HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Herr
José Ernesto Olmedo Pereira

Einführung einer flexiblen Konfigurationsfunktionalität für die GOTS GoldTrace Raw Mineral Traceability Application

Mittweida, 2018

Fakultät Angewandte Computer- und Biowissenschaften

BACHELORARBEIT

Einführung einer flexiblen Konfigurationsfunktionalität für die GOTS GoldTrace Raw Mineral Traceability Application

Autor:
Herr Jose Ernesto Olmedo Pereira

Studiengang:
Medieninformatik und Interaktives-Entertainment

Seminargruppe:
MI15w2B

Erstprüfer:
Prof. Dr.-Ing. Wilfried Schubert

Zweitprüfer:
Prof. Dr. Reinhardt Nindel

Einreichung:
Mittweida, 26.07.2018

Verteidigung/Bewertung:
Mittweida, 2018

Faculty Applied Computer Sciences &
Biosciences

BACHELOR THESIS

Introduction of a flexible configuration functionality for the GOTS GoldTrace Raw Mineral Traceability Application

author:
Mr. José Ernesto Olmedo Pereira

course of studies:
Medieninformatik und Interaktives-Entertainment

seminar group:
MI15w2B

first examiner:
Prof. Dr.-Ing. Wilfried Schubert

second examiner:
Prof. Dr. Reinhardt Nindel

submission:
Mittweida, 26.07.2018

defence/ evaluation:
Mittweida, 2018

Bibliographic Description:

Olmedo Pereira, Jose Ernesto:
Introduction of a flexible configuration functionality for the GOTS GoldTrace Raw Mineral Traceability Application. - 2018. - 3, 68, P.
Mittweida, Hochschule Mittweida, Faculty Applied Computer Sciences & Biosciences, Bachelor Thesis, 2018.

Abstract:

In this work, we discuss the key role that “conflict minerals” (Gold, Coltan, Cobalt, Tin, Tungsten) play in global supply chains and high-technology industries, and the issues surrounding their extraction and trade in origin countries, particularly in the African Congo Basin and the Great Lakes Region.

We discuss ongoing international efforts to combat violence, child labour and human rights violations at mineral extraction areas, particularly in the Democratic Republic of the Congo (DRC), where very large mineral reserves have been discovered. We present the OECD Due Diligence Guidance for Responsible Supply Chains of Minerals from Conflict-Affected and High-Risk Areas, and the GOTS MineralTrace mineral proof-of-origin and trade chain certification solution developed by ibes AG in Germany, which automates and simplifies the implementation of the OECD Guidance. We discuss a pilot project in DRC involving the GOTS GoldTrace application, based on the MineralTrace platform.

We point out MineralTrace’s benefits and its limitations. We analyse possible solutions to said limitations, including an analysis of blockchain-based transactional information exchange and record keeping systems, and finally we propose a new MineralTrace Application Programming Interface (API) that solves current limitations, introduces configuration flexibility for client applications, introduces workflow flexibility to adapt MineralTrace to any country or region, and simplifies data export functionality.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The GOTS MineralTrace platform developed by German company ibes AG, aims to help artisanal and small-scale miners (ASM) in developing countries get better prices for their production, governments to perceive urgently-needed tax revenue, and industries in importing countries to reliably audit that their supply chains conform to local and international due diligence law.

The author has been involved in the project through the development of the current MineralTrace client application, GoldTrace, which we will analyse together with the current MineralTrace API, developed by ibes AG employees based on the ICS Middleware (a proprietary software platform developed by the company). The main goal of the company while developing the MineralTrace platform is to provide law-conform traceability and due diligence mechanisms for mineral supply chains around the world. To fully achieve this goal, as we will establish after analysing the current MineralTrace system, further development must be done on both server and client-side components of the platform, in order to achieve the necessary flexibility to comply with national and international law, to quickly implement new workflows as necessary when expanding the platform to new markets, and to correctly reflect the market processes of mineral trading chains around the world.

In this work, we will propose the necessary improvements to the existing MineralTrace platform in order to achieve the full functionality envisioned by its developers. This will take the form of a new Application Programming Interface (API) that will introduce new client application configuration capabilities, new data input an export capabilities, improved identity management and new flexible workflow description mechanisms. A scientific discussion of possible solution approaches will be We will not provide a programmatic implementation of said API for the server or client-side components. Instead, we provide an API Specification in standardised notation, which can be implemented by developers of ibes AG in any programming language, operating system and environment deemed fit for each specific scenario. Therefore, we will not provide a real-world example of the new API Specification in use; since our main goal will be instead to provide the necessary technical frameworks to accurately represent national and geographical structures relevant to mineral trading chains.

Chapter 2

Basics, Status Quo and Problem Definition

The trade in natural resources can be an important source of income for developing countries, generating not only significant tax and royalty revenue streams which can be invested in the improvement of living conditions, but also a means of subsistence for millions of people around the world. In this sense, the UN Department for Economic and Social Affairs has developed a sustainable livelihood approach for artisanal mining communities [13] which has been piloted in Mali, Ethiopia, Ghana and Guinea.

The main policy recommendations for this approach, according to Labonne and Gilman (1999) are: *"Mainstreaming poverty eradication into national policymaking in all sectors including minerals"*, and *"Promoting small-scale mining as a catalyst and anchor for other productive activities to stimulate the development of complementary and alternative productive ventures necessary for sustainable poverty alleviation."* This fact helps highlight the significance of mineral resources trading around the world. One can say that tackling the issues surrounding artisanal and small-scale mining (ASM) could help improve the lives of many people around the world.

Due to their widespread usage in modern consumer products and their vital role in global trade chains, lawmakers and international institutions like the Organisation for Economic Co-operation and Development (OECD) have paid particular attention to the most commonly mined conflict minerals, since trade in those resources has fuelled armed conflict, sexual abuse and human rights violations in places like Afghanistan [1], the Central African Republic [2], and the Democratic Republic of the Congo [3]. These conflict minerals, as codified in the U.S. Conflict Minerals Law (the Dodd-Frank Wall Street Reform and Consumer Protection Act) are:

- **Columbite-tantalite** (or coltan) is the ore from which the elements Tantalum and Niobium are extracted. Tantalum is used primarily for the production of tantalum capacitors; which are known for their small compact format, high reliability and high performance. Said capacitors are used in a wide array of commonly used consumer products, from

medical devices, to airbags, Global Positioning System (GPS) and anti-locking braking systems in automobiles, through to laptop computers, smartphones, digital cameras, and more. In its carbide form, tantalum is used in jet engine/turbine blades, drill bits, end mills and other tools requiring significant hardness and wear resistance properties [4].

- **Cobalt** is used in the production of high performance metal alloys. It is also used to make rechargeable batteries, and the advent of electric vehicles and their success with consumers probably has a correlation with the Democratic Republic of Congo's soaring coltan production. Demand is expected to continue or increase as the prevalence of electric vehicles increases.[15]

- **Cassiterite** is the main ore needed to produce tin, essential for the production of tin cans and the solder used on the circuit boards of electronic equipment [5].

- **Wolframite** is an important source of the element tungsten. Tungsten is a very dense metal and is frequently used for this property, such as in fishing weights, dart tips and golf club heads. Like tantalum carbide, tungsten carbide is frequently used in applications like metalworking tools, drill bits and milling due to its considerable hardness and wear resistance properties. Smaller amounts of tungsten are used to substitute lead in "green ammunition" [6].

- **Gold** is used as currency, in jewellery, electronics, and medicaments. It is also used in certain semiconductor manufacturing processes.

Tantalum, Tin, Tungsten and Gold are sometimes referred to as "the 3TGs".

2.1 International legal status of Conflict Resources

The advancement of globalisation and international trade has lead to more and more companies outsourcing their business processes to globally dispersed production facilities. This means that social problems and human rights violations often occur in companies' supply chains, and are a significant challenge for supply chain managers [7]. Besides the negative impact conflict minerals have where they are produced, human rights violations also raise an enormous risk to corporate reputations, since the general public expects companies to behave responsibly.

Thus, companies that are located downstream in the supply chain of conflict minerals and that are more visible to consumers and regulatory agencies are particularly threatened by social supply chain problems. Conflict minerals are processed in many different components throughout various industries and hence have a high overall impact on international business.

Initiatives like the Dodd–Frank Wall Street Reform and Consumer Protection Act or the OECD Due Diligence Guidance for Responsible Supply Chains of Minerals from Conflict-Affected and High-Risk Areas [8] demand that supply chain managers verify purchased goods as "conflict-free" or implement measures to better manage any inability to do so.

The Democratic Republic of the Congo (DRC) has vast mineral resources which give it the potential to be one of Africa's richest countries. Its copper, cobalt and diamond mining industries could potentially be the largest in the continent. Therefore, this country plays a central role in the global mineral resource markets. The country is estimated to have mineral reserves including over 600 tonnes of gold, 1 billion tonnes of iron, 75 million tonnes of copper, and significant deposits of tin, cadmium, cassiterite and wolframite [9]. These facts make the country one of the world's most significant sources of 3TG minerals, and therefore a focus of attention for human rights watchdog organisations and international trade regulatory agencies. [10].

In Eastern Congo, several mining regions are affected by weak governmental presence, violence and human rights violations. Criminal armed groups are present in said areas, funded by the trade in 3TG minerals [12, pp. 21], and represent a risk to the stability of the region as well as a threat to the well-being of thousands of people. Minerals mined in this region (principally in the provinces of Maniema, South and North Kivu) pass through the hands of numerous middlemen as they are shipped out of Congo, smuggled through neighbouring countries, to East Asian processing plants [11]. In an effort to remedy this situation and promote certified conflict-free mines and trading chains, the Dodd-Frank Wall Street Reform and Consumer Protection Act applies to minerals originating (or claiming to originate) from the DRC as well as the nine adjoining countries: Angola, Burundi, Central African Republic, Congo Republic, Rwanda, South Sudan, Zimbabwe, Uganda, and Zambia.

Despite these efforts, the lack of reliable Traceability and Due Diligence mechanisms spanning whole supply chains leave downstream companies and consumers nearly incapable of detecting risks associated with conflict minerals [7]. Hence, the topic of conflict minerals becomes one of global supply chain management.

2.2 The GOTS GoldTrace Raw Mineral Traceability Application

German company ibes AG has developed the GOTS MineralTrace platform to provide law-conform traceability and due diligence mechanisms for mineral and precious stones supply chains. Said solution sets typically include secure trade documentation, end-to-end traceability and proof of origin mechanisms. The GOTS GoldTrace Raw Mineral Traceability Application (GOTS GoldTrace) is one example of an application based on the MineralTrace platform.

The Application is designed to be compliant with the OECD Due Diligence Guidance for Responsible Supply Chains of Minerals from Conflict-Affected and High-Risk Areas. According to the OECD, *"since its adoption in May 2011, the Guidance has become the leading industry standard for companies looking to live up to the expectations of the international community and customers of mineral supply chain transparency and integrity."*[1] The Guidance describes a 5-step framework applicable to both upstream (from mine to Smelter/Refiner) and downstream (from Smelter/Refiner to end consumer) supply chains which can be summarised as follows:

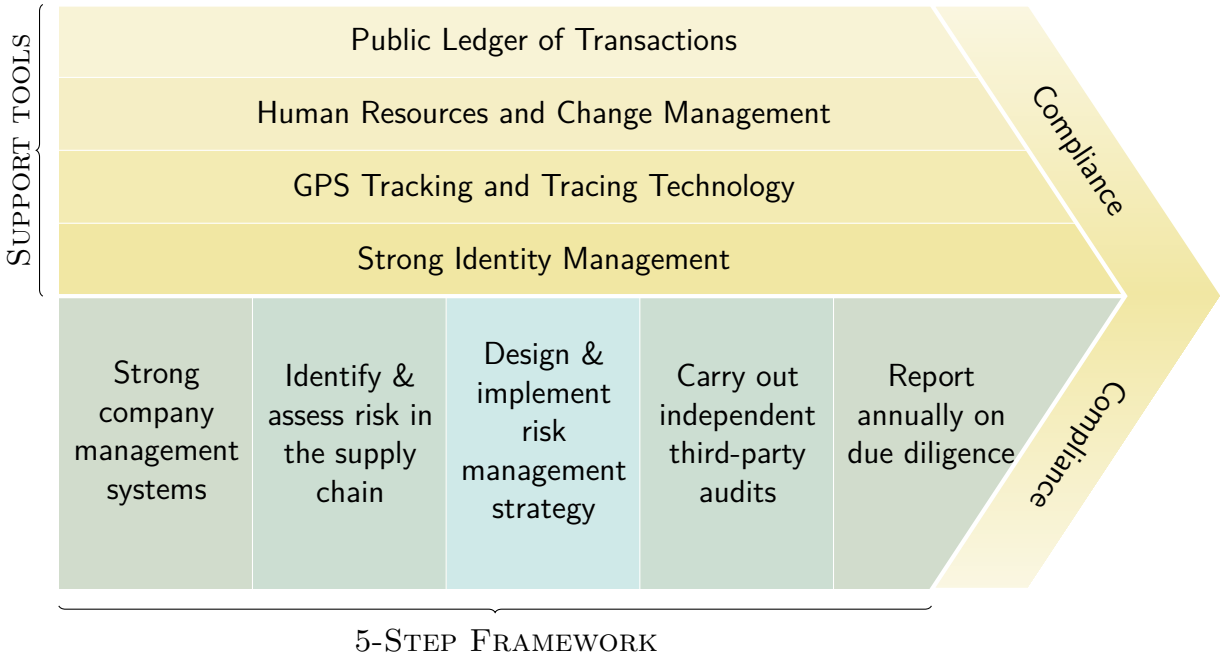


Figure 2.1: 5-Step Due Dilligence Framework including support tools as envisioned on the GOTS GoldTrace Raw Mineral Traceability Application.

[1] Cited from brochure *A GLOBAL STANDARD Towards responsible mineral supply chains*, URL: http://mneguidelines.oecd.org/Brochure_OECD-Responsible-Mineral-Supply-Chains.pdf (accessed on May 20th, 2018.)

2.2.1 Due dilligence framework for Upstream Supply Chains

The Guidance recommends the following measures that should be taken to ensure upstream supply chain compliance[2]:

1. Strong Company Management Systems:
   - Adopt a policy for responsible mineral supply chains.
   - Communicate policy to suppliers and incorporate due diligence expectations into contracts.
   - Establish traceability or chain of custody system over mineral supply chain.
2. Identify & Assess Risks in the Supply Chain:
   - Identify and verify traceability or chain of custody information (e.g. mine of origin, trade routes, suppliers).
   - For red flag locations, suppliers or circumstances, undertake on-the-ground assessments to identify risks of contributing to conflict or serious abuses.
3. Manage Risks:
   - Report identified risks to senior management and fix internal systems.
   - Disengage from suppliers associated with the most serious impacts.
   - Mitigate risk, monitor and track progress.
4. Audit of Smelter/Refiner Due Dilligence Practices:
   - Smelters/Refiners should participate in industry programmes to have their due diligence practices audited against an audit standard alligned with OECD Guidance.
   - Prepare all documentation for audit (e.g. chain of custody or traceability documentation, risk assessment and management documentation for red flagged sources).
   - Allow auditors to access company documentation and records.
   - Facilitate auditor access to sample of suppliers as appropriate.
   - Publish summary audit report with audit conclusions.
5. Publicly Report on Due Diligence:
   - Annually describe all due diligence efforts (Steps 1-4), e.g. risk assessment & mitigation, with due regard for business confidentiality and other competitive or security concerns (e.g. supplier relationships, price information, or identities of whistleblowers or sources should not be disclosed).
   - Smelters should publish a summary of their independent audit report.
   - Make report publicly available, in offices and/or the internet.

[2]Cited from infographic *OECD Due Diligence Guidance for Minerals – 5-Step Framework for Upstream and Downstream Supply Chains*,
URL: http://mneguidelines.oecd.org/5%20Step%20Framework_A3.pdf
(accessed on May 20th, 2018.)

Figure 2.2: ASM mining pits in Kampene, Maniema Province, DRC. A pilot implementation of GOTS GoldTrace was executed in the province between 2015 and 2017.

2.2.2 Application Description

GOTS GoldTrace is a software solution based on the MineralTrace platform (currently the only existing application based on it) designed to support formalisation efforts of artisanal small-scale gold mining operations and their upstream trade chains, offering OECD Guidance-conform raw material trade documentation, proof of origin and improvement of mineral royalties collection. Its main goals are to help artisanal and small-scale miners (ASM) get better prices for their production, governments to perceive urgently-needed tax revenue, and industries in importing countries to reliably audit that their supply chains conform to local and international due diligence law. The application has been piloted in the Democratic Republic of the Congo between 2015 and 2017.

☞ **Artisanal and Small-Scale Mining:** *According to* HENTSCHEL *(2003), "Broadly speaking, artisanal and small-scale mining refers to mining by individuals, groups, families or cooperatives with minimal or no mechanization [sic], often in the informal (illegal) sector of the market. Despite many attempts, a common definition of ASM has yet to be established. In some countries a distinction is made between 'artisanal mining' that is purely manual and on a very small scale, and 'small-scale mining' that has some mechanization and is on a larger scale. In some West African countries (for example, Mali), small-scale mining is differentiated from artisanal mining by the presence of permanent, fixed installations that are established once an ore body is confirmed."* [12]

Before a new MineralTrace solution can be deployed in a new country, province or region, an internationally recognised and authorised institution must independently survey the area in order to identify groups, families and/or cooperatives whose mining operations can be certified as conflict-free. These mines are then certified by said institution and recorded in the MineralTrace central database as valid, safe points of origin. Additionally, the identities of all actors authorised to participate in the upstream trade chain are recorded in the system. The following table offers a comprehensive overview of the different roles a trade chain stakeholder can take:

Role	Description
Mine pit representative	Typically, ASM sites consist of several mining pits. Co-operatives assign one of their members to represent one or more pits, and only this person is allowed to sell gold ore produced in their assigned pits.
Trader (Cat. B)	Small traders buying gold ore directly at the mining pit. Ore is then bundled and resold to cat. A traders.
Trader (Cat. A)	Large traders, Refiners and/or Smelters.
State Witness	Typically one or more state institutions have the legal mandate to supervise ASM upstream trade chains. Their agents are registered under this role.
I.D. Manager	This person is in charge of registering new stakeholders, activating I.D. cards upon delivery, and authorising I.D. card validity extensions.

Table 2.1: Possible trade chain stakeholder roles as defined in GOTS MineralTrace

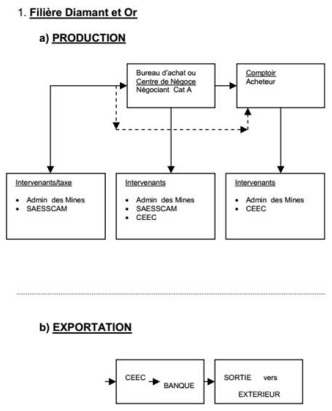


Figure 2.3: Trade actors, DRC legislation [40]

Because at the moment GoldTrace is the only MineralTrace-based application, the current role definition includes only the roles defined by DRC legislation regulating mining activities in the country, as written by the country's Ministry of Mines and Ministry of Finances [40]. The list of roles may be expanded in the future when new MineralTrace-based applications are implemented in further countries/regions or for mineral types other than gold.

GoldTrace is used directly on-site from the first purchase of gold at the mining site through resales up to the final exporter. Gold will only be transported in sealed, enumerated and electronically secured transport envelopes. For each trading lot all according transactions are recorded via the GoldTrace mobile application; assessing all relevant information per transaction, including: Seller and buyer, authenticated by personal electronic ID card, Qualifying witnesses, authenticated by personal electronic ID card, GPS position and timestamp, Transport envelope ID, Gold ore weight, Selling price.

Additional data can be entered for specific transactions, e.g. when aggregating different gold trading lots for a larger consignment or for cross-border transports. All transaction data is securely stored both on the mobile devices and in the transport envelope itself. As soon as the device has mobile data connection (GSM), it submits recorded transactions to the central GOTS online database where they are automatically checked and securely stored. All acting participants of the trade chain are able to access transaction data that they are legally authorised to see, as well as to generate reports on these transactions. Only the Ministry of Mines or equivalent state institution is allowed to access and analyse all transaction data. The GOTS GoldTrace solution is composed of several technological components, an overview of which can be seen in the diagram [2.4].

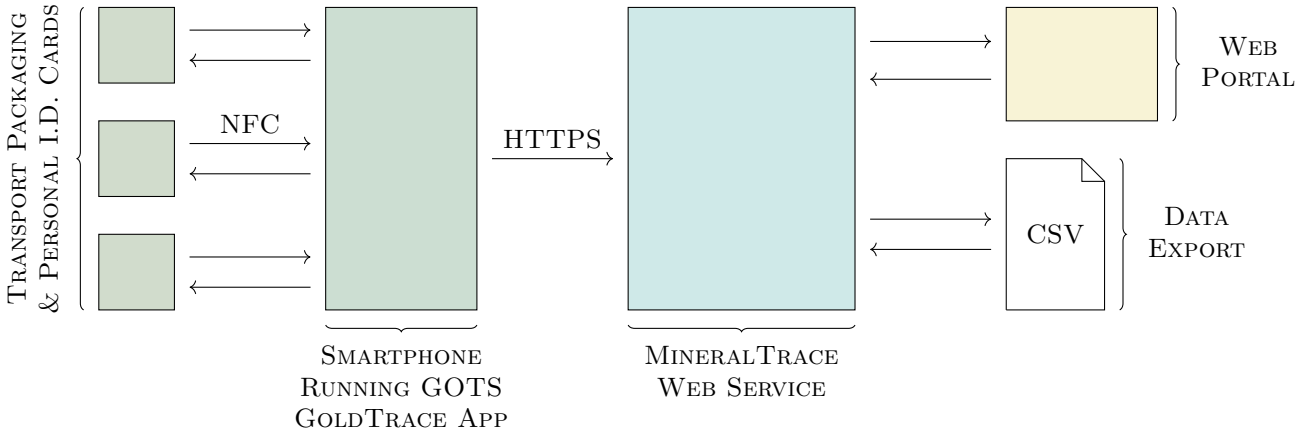


Figure 2.4: Main components of the GOTS MineralTrace solution. Web Services run on top of the ibes ICS proprietary middleware software. High-Availability, Load Balancing, Scaling, SQL Database Management, and more, are all managed by ICS.

Transport Packaging When a trader buys gold ore at a mine, the ore is placed on tamper-proof transport bags with an adhesive sealing mechanism. The bags are made from a especially formulated plastic material, which manifests colour changes when the seal or the bag itself are torn or tampered with. The transport bags include a labelling field where names and signatures can be recorded, as well as an integrated Near Field Communication (NFC) chip.

This NFC chip records encrypted transaction data of the traceability process, a unique serial number, also physically printed on the bag, which identifies the package throughout the trade chain, and a X.509 Certificate for identity representation.

There are two types of packaging: Standard, for single ore samples as sold at the mine; and Bundles, which contain inside them several Standard packages, and are typically purchased by Cat. A Traders. The weight of a standard package or bundle may never exceed 200 grams. For minerals other than gold, different packaging solutions are used (e.g. Boxes, Crates, Sacks, etc.) in order to better accommodate typical transaction volumes.

Personal I.D. Cards All trade chain stakeholders [Table 2.1] are uniquely identified through the usage of these cards. They integrate a NFC chip which records encrypted personal data, the person's role in the trade chain, the expiration date of the I.D. card, a unique serial number, which identifies the person throughout the trade chain, and a X.509 Certificate.

Lost, stolen or damaged I.D. cards have to be reported to the State institution in charge of I.D. management immediately. They will be locked immediately upon notification and may not be used anymore. The affected cardholder has to request a replacement card. Lost-and-found cards may be reactivated as long as a replacement card has not been produced and activated yet. The system will recognise any attempt to use a locked card, generate an alert, and report the case.

Smartphones Used as mobile read and write devices to interface with the NFC chips found on the transport packages and I.D. cards. The devices must have a degree of protection against intrusion (from dust, shock, accidental contact, and water) of at least IP-68, using the IP-Codes nomenclature from the IEC standard 60529 "Degrees of Protection Provided by Enclosures (IP Codes)" [14]. This IP-Code correlates to a degree of protection from solid foreign objects of "Dust tight"; and a level of protection from moisture of "Continuous Immersion". Such protection is needed due to the fact that these devices will be operated in mining environments.

The devices run the Android operating system, a MineralTrace client App, as well as a Mobile Device Management (MDM) solution to provide threat detection and removal, App blacklisting, device health monitoring, and more.

GOTS GoldTrace App Which is specifically authorised for usage and installation on a smartphone, based on said device's unique hardware identification numbers and a X.509 Client Certificate installed at the production facility where mobile devices get this mobile app pre-installed. This App is the main tool for documentation of transactions, namely:

- First sale at mining site.
- Sales between Traders.
- Bundling of envelopes and registration of transport envelopes for cross-border transports.
- Reception of transport envelopes by smelters/refiners.

The App reads data from I.D. Cards and Transport Packages; captures additional information like weight, purity and price of the ore; and obtains GPS positioning data to document transactions and check the causality of said transaction.

Checks may include but not be limited to:

- Determine if the transaction happens at an authorised mining site,
- Validate the chronological order of present and past transactions involving the package at hand,
- Check the tolerance of process-related data like variations between weight measurements, and the validity of the involved person's I.D. cards.

Given the conditions typically found in the rural areas where ASM operations take place, an important function of the App is its ability to cache transaction data when no GSM signal coverage is present and transmit it back to the MineralTrace Web Service once a data link can be established. The App runs on Android based systems.

MineralTrace Web Service Used to administer Personal I.D.s, transaction records, data reports and analyses. Access control based on Roles and unique I.D.s permits control of which specific data records are available to which actor of the process. Therefore, unauthorised access to data is prevented.

Web portal and data export facilities Are used to provide access to data to the trade chain's stakeholders, using the aforementioned access control rules. Data is available forever and never deleted. Through the Web portal, it is possible to generate the following reports:

- Persons Report: Shows the list of persons registered as stakeholders in a trading chain. The report includes the person's name, I.D. Number, Access Role, Cooperative/Mine (in the case of Mine pit representatives), and their region of work.
- Bags Report: Shows the list of transport packaging objects assigned for their use in a trading chain. The report includes the packager's I.D. Number, its Type, its Status (Created, Activated, Bundled, Ended), the manufacture and activation timestamps, and the parent package (in case of Standard packages inside a Bundling package).
- Mines Report: Displays all mines, with their corresponding mining pits, registered in a trading chain.
- Production Overview Report: Displays an overview of how much gold ore (in grams) has been sold at the mines selected in the search filter during the specified timespan.
- Production Details Report: Accessible from the production overview report, it provides the list of all transactions occurred with gold ore originating at the selected mine.
- Policy Violation: Provides a real-time view of all transactions that have failed causality checks, including the policy violation type.

All Reports can be exported to the CSV file exchange format. All client-side components of the existing GoldTrace Application were developed by the author. All server-side components of the MineralTrace platform are based on pre-existing developments of ibes AG, and the concrete implementation of the current MineralTrace API was done by ibes AG employees.

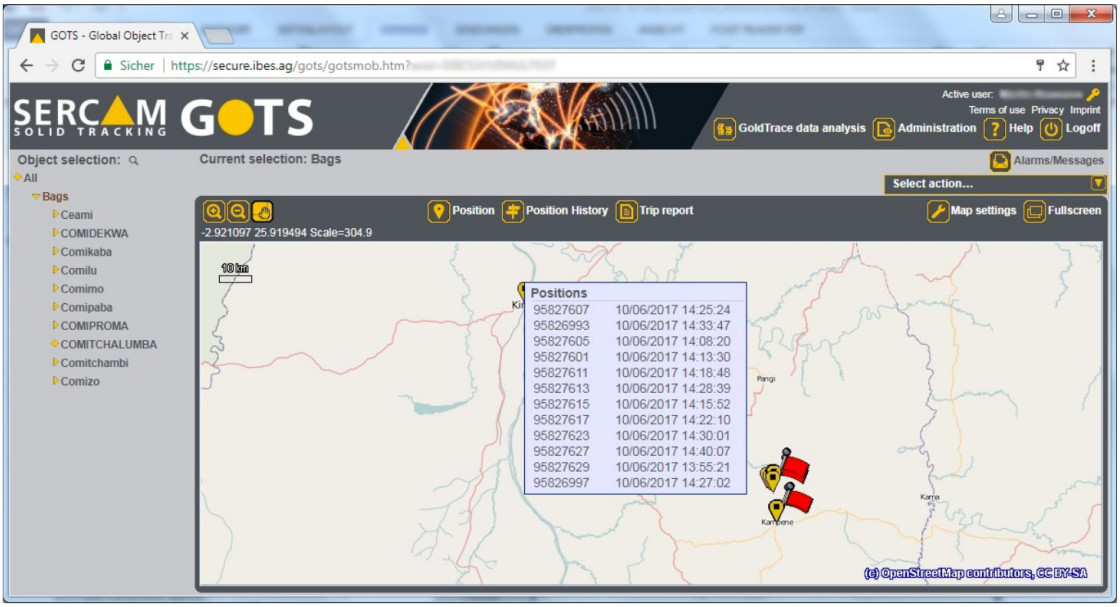


Figure 2.5: Data analysis with GOTS MineralTrace Web portal using the world map based graphical interface showing sales transactions in the area of Kampene.

A law-conform gold trading chain in DRC can be summarised in these steps:

1. Gold ore extraction at a mining pit, and sales of said ore to an authorised Trader by an authorised mine representative.
2. Further sales of gold ore between authorised Traders.
3. Bundling of several ore packages into a single transport package. A transport certificate issued by an authorised state institution is attached to the transport package, to allow lawful movement of the ore between province borders.
4. Confirmation of arrival of transport packages at a Trader Cat. A's office.
5. Smelting of gold ore, export of refined material.

Using MineralTrace, all these steps can be easily documented electronically. The use of smartphones as data capture devices allows us to take advantage of existing GSM infrastructure in the towns where miners and traders live; a clear advantage over other paper-based documentation systems.

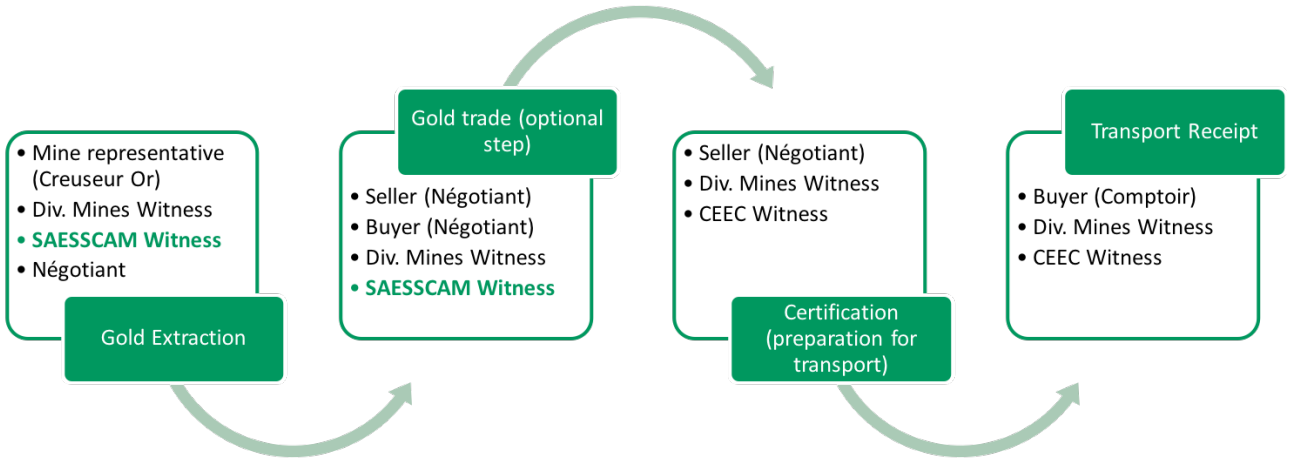


Figure 2.6: General overview of law-prescribed roles on the Gold Trade Chain in DRC and their interactions to the trade phases as documented by the GoldTrace application. These roles are registered on MineralTrace following the schema found in table [2.1]

2.3 Current MineralTrace API

The current MineralTrace API can be defined as a static set of Remote Procedure Calls reachable over HTTP GET calls. Typically, a call to the API contains the following path parameters:

- device: Device IMEI, should be pre-registered on the server.
- seccode: A Device-specific password, pre-registered on the server.
- a: Action specifier, typically the name of the RPC method.

Sample call: gti.htm?device=XXXXXXXX&seccode=XXXXXXXX&a=GETROLES
Data is exchanged as JSON Strings. Currently, the following RPCs are available:

GetProvinces: Provides a list of provinces which can be set as "Working Province" to a Person while generating said person's ID. Only provinces in the DRC are currently listed.

GetRoles: Provides a list of Roles which can be assigned to a Person.

GetMines: Provides either a list of Cooperatives, or a list of Cooperatives and their Mining Sites / Submining Sites.

GetCountries: Provides a list of registered Countries, providing their names and short ID code following the ISO-4217 standard.

GetKeystore: Provides a list of special cryptographic Salts to be used in a variety of Encryption/Decryption tasks.

GetCurrencies: Provides a list of valid currencies that can be used to express Gold Price.

GetPersons: Provides a list of Persons registered on the MineralTrace platform.

GetBags: Provides a list of transport packages registered on the MineralTrace platform.

AddID: Provides a method to register a new Person in the MineralTrace system.

ActivateID: Provides a method to activate a registered ID, used when an ID card is delivered to its owner; or when renovating an ID card.

RegBag: Provides a method to register new gold bags (small and transport) in GOTS; without assigning them to a mine.

MineSale: Through this method, the very first transaction involving a piece of gold is documented. This transaction happens at the Mine; where a Mine Representative sells gold to a "Négociant" (Trader). At this point, the gold is placed on a bag and sealed, and the bag is then bound to this gold and the mine where it comes from. Given that the required witnesses may vary, they are transmitted as an array where for each witness, their ID and role is transmitted. The server should then decide if it is satisfied with the witnesses sent, or deny the transaction if the minimum amount of witnesses is not present.

Sale: Through this method, gold buy/sell transactions between "Négotiant Cat. B" (Trader Cat. B) actors can be recorded. This transaction happens at any point between the first sale at the mine, and the bundling process. Given that the required witnesses may vary, they are transmitted as an array where for each witness, their ID and role is transmitted. The server should then decide if it is satisfied with the witnesses sent, or deny the transaction if the minimum amount of witnesses is not present.

Bundle: Through this method, small gold bags can be bundled inside a big transport bag. This process happens at the CEEC offices. A "Négotiant Cat. B" brings the small bags he wants to bundle, and receiving authorisation from a valid CEEC Transport manager, he receives a Transport Bag where he can deposit the small bags, without exceeding a total gold mass of 200g. Once bundled, the small bags get the status "2 - Bundled inside a Big Transport Bag"; and the big bag gets status "1 - Bundled, Transport Certificate present, no Transport Receipt". Given that the required witnesses may vary, they are transmitted as an array where for each witness, their ID and role is transmitted. The server should then decide if it is satisfied with the witnesses sent, or deny the transaction if the minimum amount of witnesses is not present.

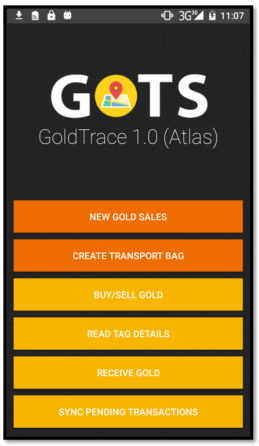
Certify: Provides a method to seal a Big Transport Bag after Small Bags have been bundled inside it. It must be bound to a Transport Certificate obtained at the CEEC Office.

Neg_A_Ack: This is the last step in the traceability chain. When the big transport bag arrives at the Trader Cat. A office (usually a smelter and/or refiner who ultimately exports the purchased gold). Upon arrival, the bag will be scanned to issue a Transport Receipt, and the bag will be marked as "Received" by the trader.

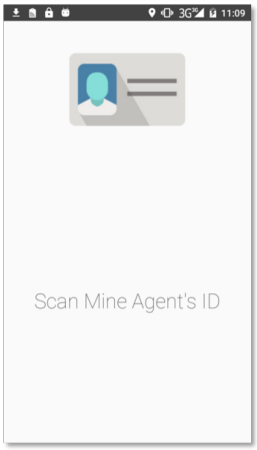
It can be seen that currently, the MineralTrace API is inflexible and highly specialised for the workflows found in the gold trade in the Democratic Republic of the Congo.

2.4 Current MineralTrace Client Implementations

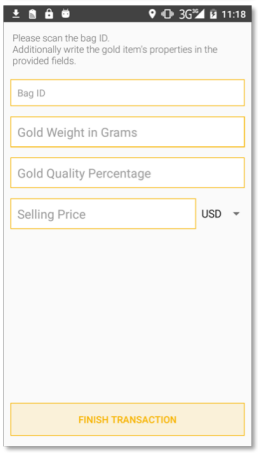
The only existing MineralTrace Client implementation at this time is the GoldTrace application used in the Democratic Republic of the Congo to document ASM gold trade. It connects to the aforementioned static RPC-based API to document transactions falling in four main categories: Initial Mine Sale, Sales between traders, Bundling of gold samples for transport, and Bundle receipt at the Trader Cat. A's office.



App entry screen with main functionality clearly visible. Additional to the four main gold trade steps as recognised by law in the DRC, the app also provides functions such as reading the RFID chips found in MineralTrace ID Cards and Packages, and Syncing pending transactions from a RFID chip or the local database to the MineralTrace backend server.



All trade actors registered on MineralTrace possess an ID card which can be scanned by the NFC sensor of the smartphones running the MineralTrace client app. Depending on the transaction type, different trade actors are mandatory. The app shows which ID should be scanned during the participant ID collection stage of transaction documentation.



After the trade participant's IDs were collected, forms with specific transaction information need to be filled. In this example, the form stating transaction details for an initial mine sale is shown. These forms are hardcoded on the Application, which means the entire application needs to be updated when a change to these forms is required. To finalise the transaction, all data is serialised, encrypted, and written to the affected package's RFID chip and sent to the backend over the Remote Procedure Call corresponding the current transaction type.

It can be seen that the current API and the client app are tightly coupled to eachother, and are static and inflexible. This means, every time a change is required, no matter how minor, code changes are needed to both server and client code. The GOTS GoldTrace solution has been piloted in gold supply chains originating in Maniema Province, DRC, from 2015 until 2017; during this time, refinements and changes were necessary to ensure proper documentation of real-world trade processes; and it was noticed that the current inflexible software architecture meant long development times before the necessary new features could go live.

The main goal of ibes AG while developing the MineralTrace platform is to provide law-conform traceability and due diligence mechanisms for mineral supply chains around the world. To fully achieve this goal, further development must be done on both server-side and client-side components of the current MineralTrace system, in order to achieve the necessary flexibility to comply with national and international law and to correctly reflect the market processes of mineral trading chains outside DRC. Currently, whenever a new kind of mineral needs to be monitored, or whenever a MineralTrace solution is needed for a new country/region, new process documentation apps need to be developed, new remote procedure calls need to be implemented on the MineralTrace server, and new reports need to be added to the MineralTrace web portal.

The product management team responsible for GOTS MineralTrace defines the following open issues:

1. The existing public MineralTrace Application Programming Interface (API) must be enhanced to allow dynamic parametrisation of trading chains (different countries, provinces, stakeholder roles, etc.); removing the need to implement new remote procedure calls and new data reports on the MineralTrace web service.

2. The new API must support dynamic parametrisation of client MineralTrace applications, enabling remote modification of workflows and rules without needing client source code changes; removing the need to develop new documentation apps.

3. The new API must remain backwards-compatible with all previous versions of it.

4. The new API must allow public access to all transaction records, while protecting private personal information and anonymising data sets where necessary. Private information must be accessible only to its owner.

Chapter 3

Solution

The current MineralTrace platform already has several advantages and benefits that make it a good compliance tool for ASM operations in Africa:

- The secure transport packaging have built-in state-of-the-art data storage, encryption and retrieval capabilities; while at the same time remaining inexpensive and robust enough to be used in rural and mining environments.
- The strong ID management mechanisms based on X.500 Directory Standards and digital signatures uncover previously unknown mineral trade stakeholders, strengthening the rule of law by giving a chance for the implementing countries to strictly enforce laws regarding mining and trade licenses, taxation, due diligence, and so on.
- By using standard rugged smartphones, the need for expensive specialised hardware is removed. Using data caching techniques, documentation apps can temporarily store encrypted transaction data until the mobile device is brought into an area with GSM network coverage. Data consistency is guaranteed by storing transaction data on the transport packages as well.

However, as previously discussed, the current platform has also aspects which can be improved. Recently, government institutions around the world have begun considering Blockchain technology as a basis for public IT services. The MineralTrace product management team has already received queries regarding the possibility of using Blockchain as basis for this improvement effort of the platform. Because of this, we shall discuss the adequacy and feasibility of Blockchain as technological basis for this platform.

3.1 Critique to currently discussed blockchain-based governmental service implementations

Crypto currencies have been conceptualised in many forms throughout the years, being no new concept in the fast-paced world of computer science and information technologies. One of the first commercial implementations of the idea of 'cryptocurrency' was DigiCash Inc's eCash system in 1990, based on two papers written by its founder (Chaum, 1983; Chaum et al., 1992) [18, 19]. However, interest in cryptocurrencies remained low until the global financial crisis of 2008. In a blog post at the beginning of the crisis, Szabo (2008) [20] argued that cryptocurrencies had the potential to counter a few problems of the fiat currency system. What Szabo described was a simple anarchist protocol that requires participants to spend resources in order to mine what he called 'digital gold' or bit gold, be rewarded for it, and in the process validate the public digital register (also known as public ledger) and achieve distributed consensus with no need for a central authority. What differentiated his approach from digital currencies of the past were the timing of the financial crisis and the distributed nature of the protocol. Rewarding the miners and the free access to the public ledger were two key innovations [29].

The first successful realisation of these ideas was developed by an anonymous person or group of persons named Satoshi Nakamoto as outlined in the October 31, 2008 whitepaper "Bitcoin: A Peer-to-Peer Electronic Cash System" [16]. This new protocol for a decentralised, peer-to-peer electronic cash system using a cryptocurrency called Bitcoin, is also the foundation of a growing number of global distributed ledgers called blockchains – of which the Bitcoin blockchain is the largest.

Recently, big banks and some governments are implementing blockchains as distributed ledgers in an attempt to innovate on the way information is stored and transactions are processed. One may argue that their interest stems from the disadvantages inherent to the prevalent fiat currency system, since fiat currencies are not completely counterfeit-secure and the anonymity of cash opens the door for informal parallel economies and black markets. Their goals might also include speed, lower costs, improved security against fraud, fewer errors, and the theoretical elimination of central points of attack and failure [17].

Upon first inspection, one may think that blockchain technology would be a perfect solution to –amongst others– supply chain and good governance challenges due to its inherent transparency. Certainly, people around the world have been proposing blockchain as a superior alternative to traditional forms of transactional information exchange and record keeping [21]. However, discussion around blockchain has mostly ignored the socioeconomic impact that widespread and indiscriminate usage of blockchain could have upon our societies and the fact that blockchain was never intended to be used for something other than being the basis for a cryptocurrency. To address these issues, we present here a short discussion of ideas requiring deeper scientific analysis.

When we consider both blockchain's technical characteristics and its fundamental anarchistic philosophy, we can identify a series of disadvantages to a state or society embracing the tech-

nology, in two separate planes: On the **technical plane**, one can identify the following issues that speak against using blockchain as the basis for widespread record keeping and transaction processing:

1. Lost transactions Dölle (2018) [25] explains how apparently confirmed transactions disappear from time to time from the Bitcoin Blockchain. Due to the way blockchain is designed to work, this is possible when single blocks or whole blockchain forks containing the affected transactions are dropped out from the long-term blockchain. In theory, this should not happen because the difficulty grade of the Proof of Work challenge is constantly adjusted to match the current total processing power of all miners inside the network. But because said difficulty grade is updated every two weeks, it can happen that whenever a significant amount of new processing power comes online, the difficulty grade becomes temporarily inadequate to prevent multiple miners finding a solution to the same challenge at the same or similar times. When that happens, the blockchain is temporarily in an undefined state, with competing groups of miners accepting one or another block. In extreme cases, the conflict is not resolved quickly enough and separate branches of the blockchain begin to form. Whenever a consensus is finally reached, all transactions containing on the losing chain of blocks get permanently dropped.[1] In non-currency applications, it is also in the realm of possibility that some transactions might get lost from the permanent public ledger. Even the possibility of losing a single transaction would be unacceptable in applications like distributed Birth or Medical Records, Traceability and Proof of Origin of natural resources, Real Estate Registrars, etc.

An alternate approach could be to manage the lifecycle of persons and objects on public IT services with digital ID certificates (ITU-T Standard X.509 v.3) following international standards like the IETF PKIX (ISO/IEC 9594-8, RFC5280). This kind of ID management is already widespread, used for example on credit card payment processing or on modern biometric passports. Storage of ID records can be achieved with public, centralised, national law-compliant X.500 directories. Said directories can implement public ID elements accessible by independent third-parties for compliance verification purposes, and private ID elements only available for responsible state authorities. The digital ID certificates can be used to digitally sign transaction records. By implementing a national public database of digitally signed transaction records, where identities are backed by the public X.500 directory, we achieve the same benefits of Blockchain without the risk of losing transactions, of conflicting or ambiguous identities, of forks and inconsistencies in the transaction chain, and so on.

2. Stuck transactions Apodaca (2017) [26] explains why confirmation of a transaction in the blockchain can become prohibitively slow: *"The most common reason for a stuck transaction is that it carries a fee that was set too low. In Bitcoin's early days, fees represented a negligible fraction of miner revenue. Today, things are very different. Fees make up an increasingly large part of total revenue, so miners try to optimise the fees they collect from every block. Pending transactions are selected by first sorting them in reverse order of fee density. Fee density (d) is*

[1] An archive of dropped blocks in the Bitcoin blockchain can be found on the website: blockchain.info/orphaned-blocks.

calculated by dividing the transaction fee (f, in satoshis[2]) by its size (s, in bytes): $d = \frac{f}{s}$ The top transaction is removed from the list and added to a candidate block. This step is repeated until the block is full. At that point any remaining transactions will need to wait for the next block to become candidates for inclusion. Most blocks are completely full today, meaning that some or even most transactions will be left behind. To a first approximation, miners don't consider a transaction's current wait time. This means that newer transactions can cut to the front of the line if they pay a higher fee density than yours." This has serious implications for public services where each and every transaction must be treated equally, which is not practicable using current blockchain technology due to its fundamental economic incentive mechanisms.

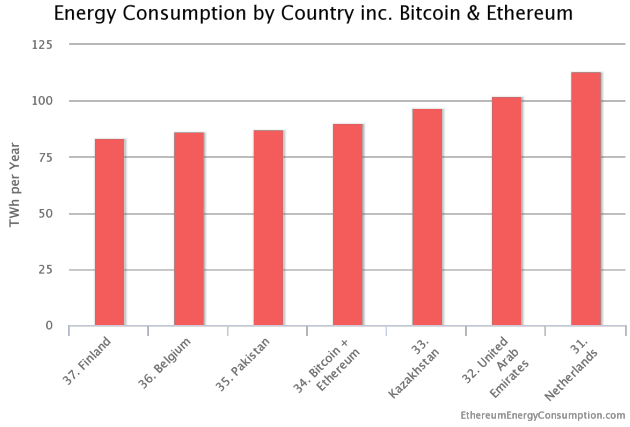


Figure 3.1: Source: Digiconomist.net.

3. Unsustainable energy consumption

The core economic mechanism of blockchain, as proposed by Nakamoto, is the Proof of Work (PoW). PoW requires miners to compete to extend the blockchain. The competition consists of solving a trivial puzzle so that success probabilities depend upon only raw computational power. A miner competes primarily through their energy expenditure, and PoW's economic incentive structure has led to an energy consumption explosion.

It can be seen on figure 3.1 that as of June 2018, Bitcoin and Ethereum together consume more energy than all but 33 countries on an annual basis. Furthermore, environmental researchers estimate that Bitcoin alone might overtake Denmark in energy consumption by 2020 [27]. If current blockchain technology were to be more widely adopted, we would soon find ourselves in an unnecessary global energy crisis with equally unnecessary damage to the environment. The blockchain community has discussed several alternatives to PoW to address this issue. The most frequently discussed alternative is the Proof of Stake (PoS) mechanism. Developers, however, remain reluctant to employ PoS because they fear that it fails to instigate consensus [28].

Compared to all known Blockchain concepts, neither stuck transactions nor unreasonable energy consumption limit the scalability of technical solutions based on X.500 Standards. The maturity and efficiency of said technology standards has been proved on a global scale by credit card payment processing systems, immigration data management systems at international borders, international banking systems, to mention a few.

On the **socioeconomic plane**, the following issues can be identified:
A core concept of modern public law is that of the monopoly of the legitimate use of force. Max Weber described in his 1919 lecture series "Politics as a Vocation" this as the defining conception of the State [22]. Weber claims that "the state is the only human Gemeinschaft (German:

[2]The satoshi is the smallest unit of the bitcoin currency recorded on the block chain. It is a one hundred millionth of a single bitcoin (0.00000001 BTC).

community) which lays claim to the monopoly on the legitimated use of physical force. However, this monopoly is limited to a certain geographical area, and in fact this limitation to a particular area is one of the things that defines a state." [23] Furthermore, the capacity of a state is often measured in terms of its ability to recover taxation to provide public goods, and its legal capacity in terms of the state's supremacy as sole arbiter of conflict resolution and contract enforcement. Without some sort of coercion, the state would be otherwise unable to enforce its legitimacy in its desired sphere of influence. [24]. We can argue that this state monopoly is crucial to maintain social order and cohesion, since modern democratic states are understood to be the sole manifestations of the will of the people living in their territories and guarantors of their human rights.

Because blockchain's premise is an ideal anarchy with full decentralisation, the state's monopoly on the use of force, and therefore its existence, is compromised: If blockchain is allowed to become a society's arbiter of (economic) conflict resolution and contract enforcement, that society becomes anarchic and its democratic power structures are replaced by new structures based on ownership and economic might whose interests, unlike those of a democratic state, are purely economic in nature and therefore with great incentive for competition, conflict and the assertive use of their (economic or physical) coercive force; and little incentive to respect the rights of their peers and the interest of the society as a whole. Losing the modern democratic state's regulatory role is something we should not accept, as this would lead to the lost of our democratic way of life, social order and cohesion.

An additional issue that arises from blockchain's distributed public ledger mechanism is that the digital self-determination rights of all stakeholders participating in a blockchain-based transactional information exchange and record keeping system are unavoidably violated. This becomes particularly relevant in the light of recent legal developments in the field of data protection and privacy rights, like the European Union's General Data Protection Regulation (GDPR) 31implemented in May 2018.

Article 17 of the GDPR (also known as the right to be forgotten), for example, is impossible to guarantee on blockchain-based solutions: No person or institution is able to delete data once it is written to the blockchain; which means digital self-determination rights cannot be respected. This would make operation and use of blockchain-based natural resource traceability systems illegal in the European Union; or at the very least, it would cause significant legal controversy.

The international community already relies on widespread use and legal regulation of digital IDs for applications like citizen registries, passports and e-commerce without any of the discussed disadvantages of Blockchain. Therefore, instead of trying to force the use of current immature blockchain technology for public services, we propose to strengthen the current legal and technological international frameworks for public registries, trade documentation, proof of origin, and other applications recently discussed as potential blockchain applications.

Loukides (2018) [30] in his article "Steering around the blockchain hype" argues that "it is certainly possible that enthusiasm for blockchains has prevented developers from investigating simpler, but less trendy, solutions." At the same time, he argues that "it's also important not to let criticism limit our imagination. When we finally find the best use cases for blockchains, they may look strange and different, like nothing we would have expected."

For the reasons analysed above, we reject the use of blockchain at its current maturity state as technological basis for public interest IT solutions like MineralTrace. We recommend gov-

ernmental institutions considering blockchain-based systems to be prudent and to not become a victim of the current hype surrounding blockchain. We propose instead to keep using internationally accepted, well-tested and well-established technologies like the alternatives mentioned above to achieve the same benefits of blockchain for the scenarios where it is currently being proposed, without the negative technical and socioeconomic impact analysed above. This approach has the additional advantage of protecting all participant's privacy and guarantees their rights for digital self-determination, which means for example that governmental institutions operating public record-keeping solutions do not need to worry about legal challenges based on the EU's GDPR inside the European Union, or similar regulations in other legislative areas.

Finally, we invite the IT industry in general to "steer around the blockchain hype" and act responsibly when deciding what technological foundations should be recommended for IT solutions in general, and public/governmental IT solutions in particular. Developers should also feel invited to conduct further research in the blockchain field. We do not deny the promising potential blockchain has to improve our communities, however we acknowledge the fact that current blockchain technology is only suitable as the basis for cryptocurrencies and not much else. We highlight the need for further scientific discussion on the technological and socioeconomic issues that widespread use of current blockchain technology can cause, with the final objective of refining or reinventing blockchain to make it feasible for applications other than cryptocurrencies and as a true way to humanely revolutionise societies, economies and governance.

3.2 Conception

3.2.1 API Information Exchange Structure

While developing a new Application Programming Interface (API), the first decision we need to take is, what communication channel to use. The MineralTrace API is consumed not only by the mobile data capturing devices, but also by end-clients (either humans using a web browser, or external web services consuming MineralTrace data for further processing). This means the chosen communication channel should be easily accessible by all communication partners over the public Internet. One possible solution would be, for example, the direct connection between peers using a custom-made communications protocol over simple Socket connections (i). Another option could be indirect communication channels like Google's Firebase Cloud Messaging (ii). A third possibility is to communicate using the HTTP/HTTPS protocol (iii).

The main advantage of using direct Socket connections (i) is that it affords total control over the communication. This is also its biggest disadvantage, because implementing a custom communications protocol is very labor-intensive and requires significant planning and testing.

Option (ii), on the other hand, would have been feasible if the API were to be consumed only by mobile end-clients. The Firebase Cloud Messaging (FCM) service was developed to simplify the transfer of messages between a server and client mobile applications. However it can also be used to send messages from the end client to the server; all the while saving battery life on the mobile client by reusing a single socket connection managed by the device's operating system. However, this kind of communication is inadequate for our purposes, since our API needs to be consumed by a heterogeneous mixture of peers.

Option (iii) using HTTP/HTTPS is also based upon Socket connections, however in this scenario, the communications protocol is already predefined. This simplifies API implementation on all ends, and by utilising TLS as implemented on HTTPS, we can expect an acceptable level of protection of our data while in transit. We have chosen this as our communication channel.

Additionally, the service structure must be defined. Here, one may either develop a completely new structure, or use well-known web service architectures like REST, SOAP, JSON-RPC, to name a few. The benefit of choosing a well-established architecture is that developers creating solutions that consume the API can more easily understand and support it.

Wagh and Thool (2012), in their article "Comparative study of soap vs rest web services provisioning techniques for mobile host." [32] provide a table comparing two of the most commonly used web service architectures, SOAP and REST. The most important differences between the two architectures are shown in the following table, which should help us make an informed decision regarding what architecture to use on our API:

SOAP	**REST**
It is well known old traditional Technology.	It is new technology as compared to SOAP.
Within the enterprise and in B2B scenarios, SOAP is still very attractive.	This is not to say that REST is not enterprise ready. In fact, there are known successful RESTful implementations in mission critical applications such as banking.

In SOAP, Client-Server interaction is tightly coupled.	In REST, Client-Server interaction is loosely coupled.
In case of implementation, SOAP overtakes REST as there are establised development kits in case of SOAP.	But REST developers would argue that it's got interface flexibility.
Changing services in SOAP web provisioning often means a complicated code change on the client side.	Changing services in REST web provisioning not requires any change in client side code.
SOAP has heavy payload as compared to REST.	REST is definitely lightweight as it is meant for lightweight data transfer over a most commonly known interface, - the URI.
It requires binary attachment parsing.	It supports all data types directly.
SOAP is not wireless infrastructure friendly.	REST is wireless infrastructure friendly.
SOAP web services always return XML data.	While REST web services provide flexibility in regards to the type of data returned.
It consumes more bandwidth because a SOAP response could require more than 10 times as many bytes as compared to REST.	It consumes less bandwidth because it's response is lightweight.
SOAP request uses POST and require a complex XML request to be created which makes response-caching difficult.	RESTful APIs can be consumed using simple GET requests, intermediate proxy servers / reverse-proxies can cache their response very easily.
Language, platform, and transport agnostic.	Language and platform agnostic.
Harder to develop, requires tools.	Much simpler to develop web services than SOAP.
False assumption: SOAP is more secure. SOAP uses WS-Security. WS-Security was created because the SOAP specification was transport-independent and no assumptions could be made about the security available on the transport layer.	REST assumes that the transport will be HTTP (or HTTPS) and the security mechanisms that are built-in to the protocol will be available.
Is the prevailing standard for web services, and hence has better support from other standards (WSDL, WS) and tooling from vendors.	Lack of standards support for security, policy, reliable messaging, etc. Services with more sophisticated requirements are harder to develop.

Because our API needs to be accessible under extremely restricted connectivity at mining sites in rural areas, a lightweight interface is indispensable. Additionally, the API needs to be loosely coupled to the clients in order to allow as most flexibility and scalability as possible. Changes to the web service should not require a change in the clients, since these are distributed around the world and client software updates will often not be a feasible option. Therefore, we have decided to implement a RESTful API.

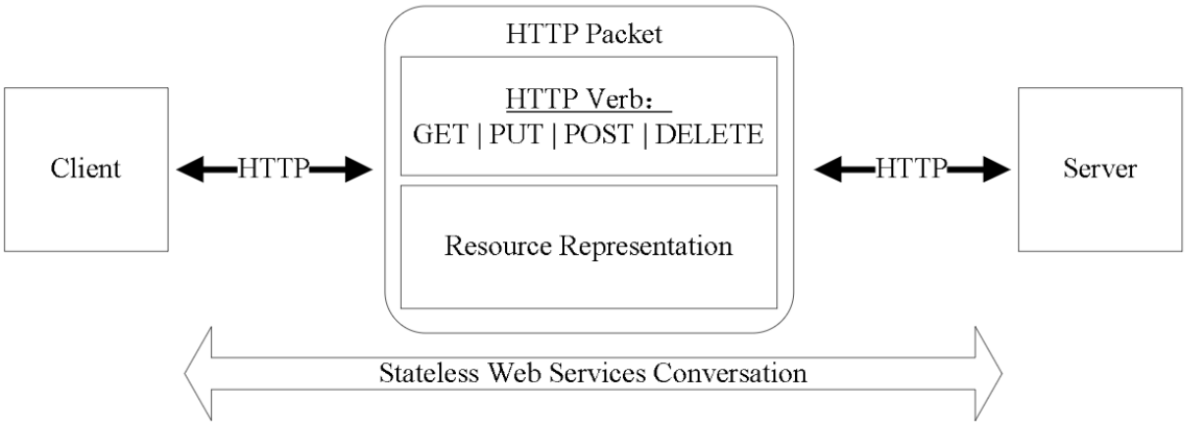


Figure 3.2: RESTful Web services architecture. Feng et al. (2009) [34]

☞**REST** (REpresentational State Transfer) is an architectural style that Roy T. Fielding first defined in his doctoral thesis "Architectural Styles and the Design of Network-based Software Architectures" [33]. Feng et al. (2009) [34] explain REST as follows: *"To understand REST, it is necessary to understand the definition of resource, representation and state. A resource can be anything. A resource may be a physical object or an abstract concept. As long as something is important enough to be referenced as a thing itself, it can be exposed as a resource. Usually a resource is something that can be stored on a computer and represented as a stream of bits. A representation is any useful information about the state of a resource. A resource may have multiple different representations. In REST there are two types of state. One is resource state which is information about a resource, and the other is application state, which is information about the path the client has taken through the application. Resource state stays on the server and application state only lives on the client. REST provides a series of architectural constraints that, when applied as a whole, emphasises scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems."*

Some of the key constraints in the REST architecture are:

1. Everything is a resource;
2. Resources are identified through URI;
3. Uniform interface;
4. Manipulation of resources through representations;
5. Self-descriptive messages;
6. Stateless interactions;
7. Hypermedia as the engine of application state.

3.2.2 Identity and Trust Management Schema

As we discussed in chapter 3.1, public infrastructure IT solutions like MineralTrace must have strong identity and trust management components, including the ability to (i) issue, manage and revoke identities at any time, (ii) have a universally trusted third-party institution as backer of identities, (iii) respect all stakeholder's privacy rights all the while (iv) strictly enforcing access rights. All these are characteristics of a Public Key Infrastructure (PKI).

☞ **PKI** is defined by Maurer (1996) [35] as *"the entire, generally heterogeneous, set of components that can be involved in issuing, storing and/or distributing certificates. A PKI can be seen as a distributed database of public-key certificates and further information (e.g. revocation lists, recommendations, etc.). It provides mechanisms for entities to retrieve and possible also to add information to the PKI. Typically, an entity Alice can retrieve another entity Bob's public key together with evidence of its authenticity. On the other hand, a user can possibly also contribute to building the infrastructure by certifying other entities' public keys or by issuing recommendations. Such certificates and recommendations can be used by entities who trust Alice but are useless for other entities (...) In a simple model of public-key certification, a user Alice uses a path (or chain) of certificates where each public key is certified by the previous entity in the path, and where she has specified the first public key as authentic and all intermediate entities as trustworthy".*

Maurer continues, *"Such a simple model can be insufficient for various reasons. First, in a realistic scenario, it should be possible to assign confidence parameters (for instance between 0 and 1) to statements about authenticity and trust. Second, it should be possible to take into account multiple certification paths which, in general, are not independent but can rather be intersecting paths in a possibly complex directed acyclic graph of certificates. Third, trust is often based on recommendations. For instance, Alice may trust an entity T she does not know because it has been recommended as trustworthy by one or several other entities that Alice trusts".* Therefore, we will attempt to correct the deficiencies pointed out by Maurer by extending existing PKI protocols where needed. This system of recommendations is of particular interest to us, because MineralTrace derives the confidence of how valid a transaction is, in part by checking the identities endorsing said transaction (traders, state witnesses and state institutions involved). Thus, the trustworthiness of signatures endorsing a transaction is directly derivable from the system's underlying structure, and cannot be easily manipulated.

One of the most widely used standards for public key certificates is the X.509 format developed by the International Telecommunications Union (ITU). It is used in major protocols and standards like TLS/SSL and HTTPS, S/MIME, EAP-TLS WiFi authentication method, IPSec, to mention a few. Therefore, we will utilise this format as our standard way of representing an identity in our API.

The structure of a X.509 certificate is expressed in a formal language, Abstract Syntax Notation One (ASN.1) [36] and can be summarised as follows:

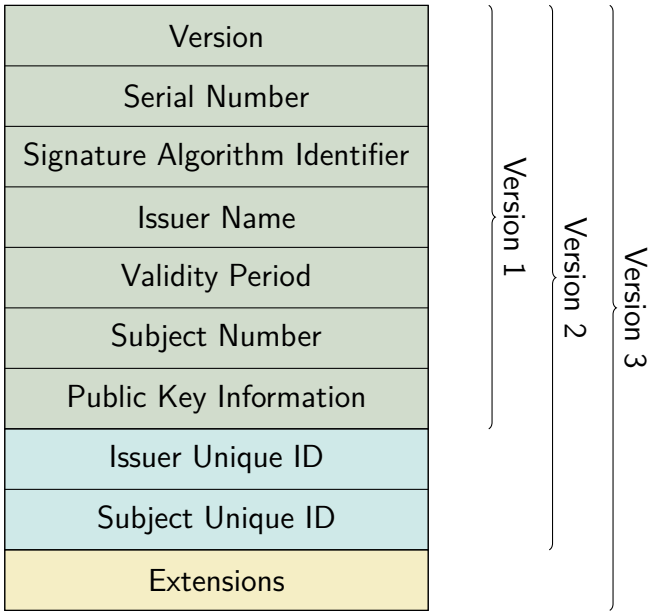


Figure 3.3: Illustration of X.509 certificate structure version 1, 2 and 3.

Our root issuing Certificate Authority will always be a trusted public institution that is recognised as trusted by all trade chain stakeholders. All device, personal and object IDs will be represented by X.509 certificates endorsed by said institution. We propose the use of X.509 version 3 certificates to take advantage of its extension fields, in order to specify additional information needed to fully represent an identity inside the MineralTrace system.

Chapter 4

Implementation

The design process of the MineralTrace API is based on the book "REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces" by Mark Massé [37].

4.1 Resource Modeling

On REST APIs, the URI path conveys the API's resource model, with each forward slash separated path segment corresponding to a unique resource within the model's hierarchy. This model is dynamic and can be changed without reprogramming the server or client software.

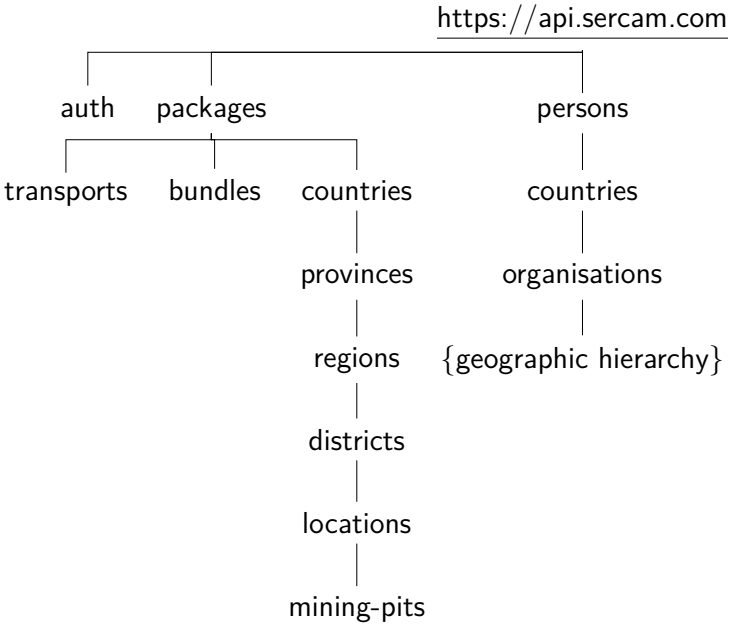


Figure 4.1: Initial MineralTrace API Resource Hierarchy. Modification is always possible.

Massé defines four main Resource Archetypes: *document, collection, store* and *controller:*

☞ **Document** resources are a concept similar to an object instance or a database record. A document's state representation typically includes both *fields* with values and *links* to other related resources. With this fundamental field and link-based structure, the document type is the conceptual base archetype of the other resource archetypes. In other words, the other three archetypes can be seen as specialisations of the document archetype.

☞ **Collection** resources are server-managed *directories* of resources. For example, the URI https://api.sercam.com/packages/countries/dr-congo represents a collection of traced objects inside the Democratic Republic of the Congo.

☞ **Store** resources are client-managed resource repositories. A store resource lets our API clients put resources in, get them back out, and decide when to update them. The example interaction below shows a package (with ID 1234) being used for the first time in a "Mine Sale" transaction at the Comilu mine in Kampene, Maniema Province, Democratic Republic of the Congo:

PUT https://api.sercam.com/packages/countries/dr-congo/provinces/maniema/regions/kampene/districts/kampene/locations/comilu/1234

☞ **Controller** resources model a procedural concept. They are like executable functions, with parameters and return values; inputs and outputs. The example interaction below shows package 1234 being used on a "Sales" transaction between two traders. Note its URI location is still "comilu", keeping a permanent relation to its place of origin:

POST https://api.sercam.com/packages/countries/dr-congo/provinces/maniema/regions/kampene/districts/kampene/locations/comilu/1234/sales

It can be seen that the API's resource hierarchy respects national and institutional hierarchies, takes due dilligence rules into consideration and aims to logically represent the relationships between each other. An advantage of using REST as architectural basis is that this hierarchy is not static but given by the URIs themselves, and thus can be updated to reflect changes in national or institutional structures without requiring code updates on the server or client components of the MineralTrace system.

4.2 API Client Authentication and Authorisation

Two important concepts that need to be clearly understood when designing any API are Client Authentication and Client Authorisation:

☞ **Authentication** refers to the verification of the credentials presented by the client during a connection attempt. Said credentials could take the form of a username/password combination, a device secret, a session token, etc.

☞ **Authorisation** is the verification that the client's connection attempt is allowed, in other words, that the client is accepted to be who it claims to be; and is allowed to conduct further transactions with the MineralTrace service.

The MineralTrace API has two authorisation scopes:

1. Limited Write Access Scope: This authorisation scope is only accessible by MineralTrace mobile devices that are pre-registered on the API's authentication service by their unique hardware identifiers. When connecting to the API, the devices present a device-specific X.509 certificate generated by the authentication service during the registration process. This certificate is installed on the client application instance running on the mobile device during the application installation and configuration process. This happens at the production facility of the mobile devices used in the MineralTrace system.
Under this scope, only **transaction** data is allowed to be posted (written) to the API, and no read operations are allowed because mobile devices are only used to document steps in the trading chain and are never used to consult reports or submit any other kind of data. When a client application wishes to log into the MineralTrace API, it submits this X.509 Certificate together with unique hardware identification numbers as credentials. The server will only accept known combinations of hardware IDs and valid Certificates, and issue a session token.

2. Read-Only Access Scope: This scope is accessibly by any user whose credentials are recognised as valid by the API's authentication service. Use cases for this scope are, for example, viewing transaction reports over a web browser, data export or consumption of MineralTrace data by external web services. In this case, credentials typically take the form of a username/-password combination; and authorisation takes the form of a session token.
Other operations, like user creation/modification/deletion, mobile device registration, or package registration, are intentionally left outside the two scopes mentioned above and not exposed over this API; because these are mission-critical operations requiring a much higher level of security and auditing than what can be achieved using a REST API.

4.2.1 JSON Web Tokens (JWT)

Authorisation grants are expressed as a session token. There are several technologies that can be used to represent said tokens; one of which are so-called JSON Web Tokens (JWTs, RFC 7519).

They are an open standard for compact, self-contained URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed or integrity protected with a Message Authentication Code (MAC) and/or encrypted. [38]

Using JWTs brings the benefit of using the same data exchange format as the rest of the API, JSON. This helps client implementations reduce the amount of code needed to parse data exchanged with the API. Additionally, by using JWS/JWE technology, we are also able to optionally digitally sign tokens, ensuring their authenticity; and encrypt them to protect the claims included inside them.

The company Auth0 Inc. has created the website https://jwt.io/introduction/ to help developers more easily understand JWTs, by presenting a condensed and more easily readable version of the RFC 7519. This subsection has been written partially based on RFC 7519 [38] and partially on the JWT.io website's content.

JSON Web Tokens consist of three parts separated by dots (.), these parts are:

- Header
- Payload
- Signature

Therefore, a JWT typically looks like this: xxxxxx.yyyyyy.zzzzzz

Header

A JWT header *typically* consists of two parts: the type of token, which is JWT, and the hashing algorithm being used. For example:

```
{
"typ" : "JWT",
"alg" : "HS256"
}
```

This JSON structure is Base64Url encoded.

Payload

This part of JWTs contains the claims. Claims are statements about an entity (for example an app instance running on a MineralTrace mobile device; or a user) and additional metadata. There are three types of claims: *registered, public,* and *private* claims.

Registered claims are a set of claims predefined on the JWT standard which are not mandatory but recommended to provide a set of useful claims. They are:

- iss (Issuer) Claim: identifies who issued the JWT. Its value is a case-sensitive string containing a StringOrURI value.

- sub (Subject) Claim: identifies who is the subject of the JWT. The subject value MUST either be scoped to be locally unique in the context of the issuer of be globally unique. Its value is a case-sensitive string containing a StringOrURI value.
- aud (Audience) Claim: identifies the recipients that the JWT is intended for. It is generally an array of case-sensitive strings, each containing a StringOrURI value.
- exp (Expiration Time) Claim: identifies the expiration time on or after which the JWT MUST NOT be accepted for processing. Its value MUST be a number containing a NumericDate value.
- nbf (Not Before) Claim: identifies the time before which the JWT MUST NOT be accepted for processing.
- iat (Issued At) Claim: identifies the time at which the JWT was issued.
- jti (JWT ID) Claim: provides a unique identifier for the JWT. The identifier value MUST be assigned in a manner that ensures that there is a negligible probability that the same value will be accidentally assigned to a different data object; if the application uses multiple issuers, collisions MUST be prevented among values produced by different issuers as well. The "jti" claim can be used to prevent the JWT from being replayed. The "jti" value is a case-sensitive string.

Public claims can be defined at will by those using JWTs. But to avoid collisions they should be defined in the IANA JSON Web Token Registry or be defined as a URI that contains a collision resistant namespace.

Private claims are the custom claims created to share information between parties that agree on using them and are neither registered or public claims.

An example payload of a token issued to a user browsing transaction data through a web browser could be:

```
{
    "sub" : "jdoe",
    "iss" : "A123456789",
    "exp" : 123456789,
    "jti" : "B987654321"
}
```

The payload is Base64Url encoded to form the second part of a JWT.

Signature

The signature part of a JWT is created by taking the encoded header, the encoded payload, a secret, the algorithm specified in the header, and signing that. For example, if we were to choose the HMAC SHA256 algorithm, the signature would be created in this fashion:

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

The signature is used to verify the message wasn't changed along the way, and, in the case of tokens signed with a private key, it can also verify that the sender of the JWT is who it says it is.

4.3 API Specification

In this section, we will put together all concepts discussed thus far, and provide a specification of the MineralTrace API version 2. To aid the API design process, the Swagger[1] Editor tool developed by SmartBeat Software was used.

☞ **How to read this specification:** Swagger allows us to design, describe and document APIs using the OpenAPI[2] notation, we used this notation to describe our API using a standardised format understandable to both humans and machine parsers.

4.3.1 Data Models

The following data models are proposed to facilitate communication between clients and the API. Because of the open nature of the MineralTrace API, however, more data models can be added in the future. Objects are specified using the OpenAPI notation.

Listing 4.1: This model is used on the "/" URI path. It allows a client to quickly get a reference of all API resources under the root document.

```
Root:
  properties:
    links:
      type: array
      description: Lists all possible collections and controllers under
         this node in the API URI hierarchy.
      items:
        type: string
```

Listing 4.2: Place models are used whenever a representation for a Country, Province, Region, District, Location, Mining Pit, or any other fixed geographical location is required.

```
PlaceArray:
  properties:
    currentConfigVersion:
```

[1] https://swagger.io/tools/swagger-editor/
[2] https://swagger.io/specification/

```
        type: integer
        format: int32
      places:
        type: array
        items:
          $ref: '#/components/schemas/Place'
      links:
        type: array
        description: Lists all possible collections and controllers under
           this node in the API URI hierarchy
        items:
          type: string
  PlaceType:
    properties:
      id:
        type: integer
        format: int32
      hierarchyIndex:
        type: integer
        format: int32
      subordinateHierarchyIndexes:
        type: array
        items:
          type: integer
      name:
        type: object
        $ref: '#/components/schemas/LocalisedString'
  Place:
    properties:
      id:
        type: integer
        format: int32
      type:
        type: object
        $ref: '#/components/schemas/PlaceType'
      parentPlace:
        type: integer
        format: int32
      childrenPlaces:
        type: array
        items:
          $ref: '#/components/schemas/Place'
      apiName:
        type: string
```

```
      latitude:
        type: number
        format: double
      longitude:
        type: number
        format: double
      name:
        type: array
        items:
          $ref: '#/components/schemas/LocalisedString'
```

Listing 4.3: Because MineralTrace operates in several countries around the world, a flexible currency representation format is mandatory.

```
  Currency:
    properties:
      isoCode:
        type: string
      name:
        type: string
```

Listing 4.4: Every time a 'package' resource is manipulated, we need the PackageResponse data model to deliver back to the client a representation of the current state of the packages involved in the original query.

```
  PackageResponse:
    properties:
      currentConfigVersion:
        type: integer
        format: int32
      foundPackages:
        type: array
        items:
          $ref: '#/components/schemas/Package'
      links:
        type: array
        description: Lists all possible collections and controllers under
           this node in the API URI hierarchy
        items:
          type: string
```

Listing 4.5: Because MineralTrace apps need to support multiple languages, all text strings coming to/from the API, which are visible to the user, need multilanguage support.

```
  LocalisedString:
    properties:
```

```
      languageIsoCode:
        type: string
      value:
        type: string
```

Listing 4.6: Whenever a query involving transactions is executed, we need the TransactionResponse data model to deliver back to the client a representation of the transactions involved in the original query.

```
  TransactionResponse:
    properties:
      currentConfigVersion:
        type: integer
        format: int32
      foundTransactions:
        type: array
        items:
          $ref: '#/components/schemas/Transaction'
      links:
        type: array
        description: Lists all possible collections and controllers under
           this node in the API URI hierarchy
        items:
          type: string
```

Listing 4.7: Because we want to make MineralTrace Apps as flexible as possible, there are no fixed transaction types on the applications themselves. Instead, they consult the API which transaction types are applicable for the country and mineral type relevant to them.

```
  TransactionType:
    properties:
      typeId:
        type: integer
        format: int32
      typeName:
        type: object
        $ref: '#/components/schemas/LocalisedString'
```

Listing 4.8: Representation of a Transaction as delivered to a client.

```
  Transaction:
    properties:
      id:
        type: integer
        format: int64
      type:
```

```
        type: object
        $ref: '#/components/schemas/TransactionType'
      seller:
        type: object
        $ref: '#/components/schemas/Person'
      buyer:
        type: object
        $ref: '#/components/schemas/Person'
      witnesses:
        type: array
        items:
          $ref: '#/components/schemas/Person'
      tradedPackages:
        type: array
        items:
          $ref: '#/components/schemas/Package'
      transactionTime:
        type: string
        format: date-time
      transactionLongitude:
        type: number
        format: double
      transactionLatitude:
        type: number
        format: double
      deviceUsed:
        type: object
        $ref: '#/components/schemas/MobileDevice'
      extras:
        type: object
```

Listing 4.9: Representation of a mobile device running a MineralTrace application. For statistical and law enforcement purposes, it is important to always be able to determine who owns a device.

```
  MobileDevice:
    properties:
      deviceId:
        type: string
      owner:
        type: object
        $ref: '#/components/schemas/Person'
```

Listing 4.10: There are no fixed package types or status hardcoded on the MineralTrace apps. Instead, apps query the API to see what package types and status are applicable to the country, mineral type and workflow they are intended for. Using the AppConfig 4.20 object.

```
PackageStatus:
  properties:
    applicableInCountry:
      type: object
      $ref: '#/components/schemas/Place'
    name:
      type: object
      $ref: '#/components/schemas/LocalisedString'
    statusId:
      type: integer
    applicableForPackageType:
      type: object
      $ref: '#/components/schemas/PackageType'
PackageType:
  properties:
    type:
      type: string
      enum:
        - transportPackage
        - bundlePackage
        - smeltPackage
    name:
      type: object
      $ref: '#/components/schemas/LocalisedString'
    apiURI:
             type: string
    applicableMineral:
      type: object
      $ref: '#/components/schemas/MineralType'
    applicableCountry:
      type: object
      $ref: '#/components/schemas/Place'
PackageTypeResponse:
  properties:
    currentConfigVersion:
      type: integer
      format: int32
    typeDefinition:
      type: array
      items:
```

```
          $ref: '#/components/schemas/PackageType'
      links:
        type: array
        description: Lists all possible collections and controllers under
           this node in the API URI hierarchy
        items:
          type: string
```

Listing 4.11: Representation of a package object.

```
  Package:
    properties:
      packageId:
        type: string
      packageType:
        type: object
        $ref: '#/components/schemas/PackageType'
      mineralType:
        type: object
        $ref: '#/components/schemas/MineralType'
      weight:
        type: object
        $ref: '#/components/schemas/Weight'
      quality:
        type: number
        format: int32
      price:
        type: object
        $ref: '#/components/schemas/Price'
      status:
        type: object
        $ref: '#/components/schemas/PackageStatus'
      origin:
        type: object
        $ref: '#/components/schemas/Place'
      sealingTime:
        type: string
        format: date-time
      sealingLatitude:
        type: number
        format: double
      sealingLongitude:
        type: number
        format: double
      totalPacksIncluded:
```

```
        type: integer
        format: int32
      includedPackageIds:
        type: array
        items:
          type: string
      transportCertificateId:
        type: string
      transportProvider:
        type: string
      transportProviderNumber:
        type: string
      crossing:
        type: object
        $ref: '#/components/schemas/Place'
      destinationCity:
        type: object
        $ref: '#/components/schemas/Place'
      destinationCountry:
        type: object
        $ref: '#/components/schemas/Place'
```

Listing 4.12: When an application wishes to log into the API, it follows a special procedure where it submits its X.509 Digital Certificate as credentials, as well as its hardware identifiers. See 4.2.

```
  AppAuthRequest:
    properties:
      x-509-cert:
        type: string
      appversion:
        type: integer
        format: int32
      appid:
        type: string
      osversion:
        type: string
      serial:
        type: string
```

Listing 4.13: Sometimes, not all stakeholders holding valid trading licenses are known to the MineralTrace system. When this happens, and if the Workflow rules allow it, it is possible to manually input the identity of a trade participant using the PersonOverrideObject.

```
  PersonOverrideObject:
    properties:
```

```
      role:
        type: object
        $ref: '#/components/schemas/Role'
      names:
        type: array
        items:
          type: object
          properties:
            index:
              type: integer
            name:
              type: string
      legalIdNumber:
        type: string
      legalIdIssuer:
        type: string
      contactData:
        type: array
        items:
          type: string
```

Listing 4.14: We could in theory run all transaction postings to the API using a single abstract object. However, one of the goals of a RESTful API is to be as clear to the developer as possible. Therefore, we have added convenience controller resources to the package URIs to easily implement transaction API calls.

```
  MineSaleRequest:
    properties:
      sealingLatitude:
        type: number
        format: double
      sealingLongitude:
        type: number
        format: double
      sealingTime:
        type: string
        format: date-time
      sellingPrice:
        type: object
        $ref: '#/components/schemas/Price'
      origin:
        type: object
        $ref: '#/components/schemas/Place'
      seller:
        type: object
```

```
        $ref: '#/components/schemas/Person'
      buyer:
        type: object
        $ref: '#/components/schemas/Person'
      weight:
        type: object
        $ref: '#/components/schemas/Weight'
      quality:
        type: number
        format: double
      witnesses:
        type: array
        items:
          $ref: '#/components/schemas/Person'
      overridePersons:
        type: array
        items:
          $ref: '#/components/schemas/PersonOverrideObject'
  SaleRequest:
    properties:
      saleLatitude:
        type: number
        format: double
      saleLongitude:
        type: number
        format: double
      saleTime:
        type: string
        format: date-time
      salePrice:
        type: object
        $ref: '#/components/schemas/Price'
      seller:
        type: object
        $ref: '#/components/schemas/Person'
      buyer:
        type: object
        $ref: '#/components/schemas/Person'
      weight:
        type: object
        $ref: '#/components/schemas/Weight'
      witnesses:
        type: array
        items:
```

```
          $ref: '#/components/schemas/Person'
      overridePersons:
        type: array
        items:
          $ref: '#/components/schemas/PersonOverrideObject'
  BundleRequest:
    properties:
      totalBundlePrice:
        type: object
        $ref: '#/components/schemas/Price'
      totalPacksIncluded:
        type: integer
        format: int32
      sealingLatitude:
        type: number
        format: double
      sealingLongitude:
        type: number
        format: double
      sealingTime:
        type: string
        format: date-time
      totalBundleWeight:
        type: object
        $ref: '#/components/schemas/Weight'
      witnesses:
        type: array
        items:
          $ref: '#/components/schemas/Person'
      trader:
        type: object
        $ref: '#/components/schemas/Person'
      includedPackageIds:
        type: array
        items:
          type: string
      transportCertificateId:
        type: string
      transportProvider:
        type: string
      transportProviderNumber:
        type: string
      crossing:
        type: object
```

```
        $ref: '#/components/schemas/Place'
      destinationCity:
        type: object
        $ref: '#/components/schemas/Place'
      destinationCountry:
        type: object
        $ref: '#/components/schemas/Place'
      overridePersons:
        type: array
        items:
          $ref: '#/components/schemas/PersonOverrideObject'
  ReceiptRequest:
    properties:
      totalBundlePrice:
        type: object
        $ref: '#/components/schemas/Price'
      totalPacksIncluded:
        type: integer
        format: int32
      receiptLatitude:
        type: number
        format: double
      receiptLongitude:
        type: number
        format: double
      receiptTime:
        type: string
        format: date-time
      totalBundleWeight:
        type: object
        $ref: '#/components/schemas/Weight'
      witnesses:
        type: array
        items:
          $ref: '#/components/schemas/Person'
      receiver:
        type: object
        $ref: '#/components/schemas/Person'
      overridePersons:
        type: array
        items:
          $ref: '#/components/schemas/PersonOverrideObject'
```

Listing 4.15: Representation of a trade stakeholder.

```
  Person:
    properties:
      personId:
        type: string
      role:
        type: object
        $ref: '#/components/schemas/Role'
      firstName:
        type: string
      lastName:
        type: string
      organisation:
        type: object
        $ref: '#/components/schemas/Organisation'
      workingTerritory:
        type: object
        $ref: '#/components/schemas/Place'
      idCardValidUntil:
        type: string
        format: date-time
      idCardActivated:
        type: string
        format: date-time
  PersonResponse:
    properties:
      currentConfigVersion:
        type: integer
        format: int32
      foundPersons:
        type: array
        items:
          $ref: '#/components/schemas/Person'
      links:
        type: array
        description: Lists all possible collections and controllers under
           this node in the API URI hierarchy
        items:
          type: string
```

Listing 4.16: A stakeholder with the appropriate management role assigned is able to activate and renovate personal ID cards of other MineralTrace stakeholders.

```
  ActivateIdRequest:
    properties:
```

```
      personId:
        type: string
      managerId:
        type: string
      activationLatitude:
        type: number
        format: double
      activationLongitude:
        type: number
        format: double
      activationTime:
        type: string
        format: date-time
      validPeriodInSeconds:
        type: integer
        format: int64
```

Listing 4.17: A Stakeholder is always considered a member of either a state or private organisation.

```
  OrganisationType:
    properties:
      typeId:
        type: integer
        format: int32
      type:
        type: string
        enum:
          - stateInstitution
          - smelter
          - mineOrCooperative
          - trader
      place:
        type: object
        $ref: '#/components/schemas/Place'
  OrganisationResponse:
    properties:
      currentConfigVersion:
          type: integer
          format: int32
      organisations:
        type: array
        items:
          $ref: '#/components/schemas/Organisation'
      links:
        type: array
```

```
        description: Lists all possible collections and controllers under
           this node in the API URI hierarchy
        items:
          type: string
  Organisation:
    properties:
      organisationId:
        type: integer
        format: int32
      name:
        type: object
        $ref: '#/components/schemas/LocalisedString'
      type:
        type: object
        $ref: '#/components/schemas/OrganisationType'
```

Listing 4.18: When a user logs in to the read-only API access scope, they obtain a JWT to authenticate their further queries to the API.

```
  Auth:
    required:
      - id
      - jwt
    properties:
      id:
        type: integer
        format: int64
      jwt:
        type: string
```

Listing 4.19: Apps log in to the limited-write API access scope, and therefore they obtain further data relevant to them. The most important parameter a MineralTrace app gets when logging in to the API is the currentConfigVersion. This tells the app if the configuration version it has stored locally is outdated and needs to be refreshed.

```
  AuthApp:
    required:
      - id
      - jwt
    properties:
      id:
        type: integer
        format: int64
      jwt:
        type: string
      currentConfigVersion:
```

```
        type: integer
        format: int32
      links:
        type: array
        description: Lists all possible collections and controllers under
           this node in the API URI hierarchy
        items:
          type: string
```

Listing 4.20: Application configuration object, which allows fully dynamic configuration of all workflow steps, fields for each step, rules for required stakeholders, tolerances for data input fields, and more. **The AppConfig object is the key element to fulfil our goal to introduce a flexible configuration functionality for all MineralTrace apps including GoldTrace.**

```
  AppConfig:
    properties:
      configVersion:
        type: integer
        format: int32
      workflowSteps:
        type: array
        items:
          $ref: '#/components/schemas/WorkflowStep'
      toleranceParameters:
        type: object
        $ref: '#/components/schemas/Tolerance'
      appMineralType:
        type: object
        $ref: '#/components/schemas/MineralType'
      targetCountry:
        type: object
        $ref: '#/components/schemas/Place'
  Tolerance:
    properties:
      referenceWeights:
        type: array
        items:
          $ref: '#/components/schemas/ReferenceWeight'
      referencePrices:
        type: array
        items:
          $ref: '#/components/schemas/ReferencePrice'
  ReferenceWeight:
    properties:
```

```
      policyId:
        type: integer
        format: int32
      mineralType:
        type: object
        $ref: '#/components/schemas/MineralType'
      minimumWeightTransport:
        type: object
        $ref: '#/components/schemas/Weight'
      maximumWeightTransport:
        type: object
        $ref: '#/components/schemas/Weight'
      minimumWeightBundle:
        type: object
        $ref: '#/components/schemas/Weight'
      maximumWeightBundle:
        type: object
        $ref: '#/components/schemas/Weight'
  ReferencePrice:
    properties:
      policyId:
        type: integer
        format: int32
      mineralType:
        type: object
        $ref: '#/components/schemas/MineralType'
      referencePrice:
        type: object
        $ref: '#/components/schemas/Price'
      perUnitOfMeasure:
        type: object
        $ref: '#/components/schemas/WeightUnit'
      maxDeviation:
        type: number
        format: double
  WorkflowStep:
    properties:
      index:
        type: integer
        format: int32
      transactionType:
        type: object
        $ref: '#/components/schemas/TransactionType'
      displayName:
```

```
        type: array
        items:
          $ref: '#/components/schemas/LocalisedString'
      idCardRolesRequired:
        type: array
        items:
          type: object
          $ref: '#/components/schemas/WorkflowRoleDescription'
      workflowFieldsEnabled:
        type: array
        items:
          type: object
          $ref: '#/components/schemas/WorkflowFieldDescription'
      allowableInputStatus:
        type: array
        items:
          $ref: '#/components/schemas/PackageStatus'
      setOutputStatus:
        type: array
        items:
          $ref: '#/components/schemas/PackageStatus'
      inputPackageType:
        type: array
        items:
          $ref: '#/components/schemas/PackageType'
      outputPackageType:
        type: array
        items:
          $ref: '#/components/schemas/PackageType'
  WorkflowFieldDescription:
    properties:
      fieldType:
        type: string
        enum:
          - number
          - currency
          - weight
          - text
          - person
      roleIndex:
        type: integer
        format: int32
      isMandatory:
        type: boolean
```

```
      fieldId:
        type: string
      index:
        type: integer
        format: int32
      parentWorkflow:
        type: integer
        format: int32
      isEnabled:
        type: boolean
      applyTolerancePolicy:
        type: boolean
      tolerancePolicyId:
        type: integer
        format: int32
  WorkflowRoleDescription:
    properties:
      index:
        type: integer
        format: int32
      role:
        type: object
        $ref: '#/components/schemas/Role'
      isManualOverwriteAllowed:
        type: boolean
```

Listing 4.21: Utility data models useful to represent commonly used data.

```
  Price:
    properties:
      value:
        type: number
        format: double
      currency:
        type: object
        $ref: '#/components/schemas/Currency'
  Weight:
    properties:
      value:
        type: number
        format: double
      unit:
        type: object
        $ref: '#/components/schemas/WeightUnit'
  WeightUnit:
```

```
    type: string
    enum:
      - gram
      - fra
  MineralType:
    type: string
    enum:
      - gold
      - coltan
      - cobalt
      - cassiterite
      - wolframite
  Role:
    properties:
      roleId:
        type: integer
        format: int32
      displayName:
        type: array
        items:
          $ref: '#/components/schemas/LocalisedString'
  Error:
    required:
      - code
      - message
    properties:
      code:
        type: integer
        format: int32
      message:
        type: string
```

4.3.2 URI Resources

On the digital version of this document, please open the HTML file

```
./dist/index.html
```

on the browser of your choosing to see an interactive version of the MineralTrace API documentation. Alternatively, the author of this document can be contacted via E-Mail (jolmedo@hs-mittweida.de) with a request to obtain a copy of this interactive documentation. Based on the Resource Hierarchy (See Figure 4.1) proposed at the beginning of this chapter, we can identify the following API resources:

General Resources

GET / Root Document, delivers a list of links to underlying resources. This helps clients to quickly get oriented on the API and find available resources.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}
```

Response 200: an array of links to the available collections, documents and controllers under the root document. [See Root object on listing 4.1]
Default response: unexpected error. [See Error object on listing 4.21].

Authentication Resources

GET /auth/loginUser Logs in a user in read-only mode.

```
Parameters:
    - Query:
        username (String)

        password (String): The MD5-hashed user password.
```

Response 200: a JSON Web Token representing authorisation. [See Auth object on listing 4.18]
Response 400: invalid username/password. [See Error object on listing 4.21].
Default response: unexpected error. [See Error object on listing 4.21].

POST /auth/loginApp Logs in an app in limited-write mode.

```
Parameters:
    - Query:
        imei (String): Device's primary IMEI number.

    - Body: Application identifiers including X.509 certificate specific for
        device/app combination (application/json), example:
```

```
{
    "x-509-cert": "string",
    "appversion": 0,
    "appid": "string",
    "osversion": "string",
    "serial": "string"
}
```

Response 200: a JSON Web Token representing authorisation, the app configuration version available on the server, and a "links" object specifying available resources under the current path. [See AuthApp object on listing 4.19]
Response 400: invalid username/password. [See Error object on listing 4.21].
Default response: unexpected error. [See Error object on listing 4.21].

GET /auth/loginApp/getConfig If an app determines its local configuration is older than the one available on the server, or if no configuration is present, it calls this resource to get a current configuration. **The configuration object returned under this Controller is the key to fulfil our goal to introduce a flexible configuration functionality for all MineralTrace apps including GoldTrace.**

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Query:
        configVersion (Integer): Config version the app is asking for.

        forApp (Integer): Application Package ID.
```

Response 200: the app configuration version requested. [See AppConfig object on listing 4.20].
Default response: unexpected error. [See Error object on listing 4.21].

Person Resources

GET /persons Persons collection, list of all persons registered on MineralTrace.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Query:
        scope (String): How deep the query should run in LDAP search scope
           format. Available values : base, one, sub
```

Response 200: an array of Person objects [See PersonsResponse object on listing 4.15].
Default response: unexpected error. [See Error object on listing 4.21].

Further person collections can be found under this path: For example, we can also list persons registered in a country (/persons/countries/{country-name}/), a province, a district, etc. The response would be a PersonsResponse object [Listing 4.15].

GET /persons/{person-id} Person document, delivers details of the specified person.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        person-id (String): ID of the desired person.
```

Response 200: an array of Person objects, with one element: the specified person [See PersonsResponse object on listing 4.15].
Default response: unexpected error. [See Error object on listing 4.21].

POST /persons/{person-id}/activate Controller used to activate or renovate the specified person's MineralTrace ID Card.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        person-id (String): ID of the desired person.

    - Body: a Person object (application/json), example:
        {
            "personId": "string",
            "managerId": "string",
            "activationLatitude": 0,
            "activationLongitude": 0,
            "activationTime": "2018-06-25T15:35:50.772Z",
            "validPeriodInSeconds": 0
        }
```

Response 200: an array of Person objects, with one element: the specified person [See PersonsResponse object on listing 4.15].
Default response: unexpected error. [See Error object on listing 4.21].

GET /persons/{person-id}/transactions Collection used to get all transactions where the specified person participated.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}
```

```
  - Query:
      scope (String): How deep the query should run in LDAP search scope
         format. Available values : base, one, sub

      type (Array): Allows to optionally specify what transaction types to
         list.

  - Path:
      person-id (String): ID of the desired person.
```

Response 200: a list of transaction objects [See TransactionsResponse object on listing 4.6].
Default response: unexpected error. [See Error object on listing 4.21].

GET /persons/countries List of all countries recognised by MineralTrace.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}
```

Response 200: an array of Country objects [See PlaceArray object on listing 4.2].
Default response: unexpected error. [See Error object on listing 4.21].
Further place collections can be found under this path: For example, we can also list provinces registered in a country (/persons/countries/{country-name}/provinces), a districts in a province, and so on. The response would be a PlaceArray object [Listing 4.2].

GET /persons/countries/{country-name}/organisations List of all organisations registered on MineralTrace for the specified country.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}
    - Path:
        country-name (String): API name of the country of interest.
```

Response 200: an array of Organisation objects [See OrganisationResponse object on listing 4.17].
Default response: unexpected error. [See Error object on listing 4.21].

Package Resources

GET /packages List of all packages tracked by MineralTrace.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}
```

```
    - Query:
        scope (String): How deep the query should run in LDAP search scope
           format. Available values : base, one, sub
```

Response 200: an array of MineralTrace packages [See PackageResponse object on listing 4.4].

Default response: unexpected error. [See Error object on listing 4.21].

Further package collections can be found under this path: For example, we can also list packages registered in a country (/packages/countries/{country-name}/), a province, a district, etc. The response would be a PackageResponse object [Listing 4.4].

GET /packages/transactions Get all transactions recorded on MineralTrace.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Query:
        scope (String): How deep the query should run in LDAP search scope
           format. Available values : base, one, sub

        type (Array): Allows to optionally specify what transaction types to
           list.
```

Response 200: a list of transaction objects [See TransactionsResponse object on listing 4.6].

Default response: unexpected error. [See Error object on listing 4.21].

Further transaction collections can be found under this path: For example, we can also list transactions registered in a country (/packages/countries/{country-name}/transactions), a province, a district, etc. The response would be a TransactionsResponse object [Listing 4.6].

GET /packages/{package-id} Get a specific package by ID.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        package-id (String): Desired package's ID.
```

Response 200: an array of MineralTrace packages, including only the desired package [See PackageResponse object on listing 4.4].

Default response: unexpected error. [See Error object on listing 4.21].

GET /packages/{package-id}/transactions Get all transactions where this package was involved.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Query:
        scope (String): How deep the query should run in LDAP search scope
           format. Available values : base, one, sub

        type (Array): Allows to optionally specify what transaction types to
           list.

    - Path:
        package-id (String): Desired package's ID.
```

Response 200: a list of transaction objects [See TransactionsResponse object on listing 4.6].
Default response: unexpected error. [See Error object on listing 4.21].

GET /packages/countries List of all countries recognised by MineralTrace.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}
```

Response 200: an array of Country objects [See PlaceArray object on listing 4.2].
Default response: unexpected error. [See Error object on listing 4.21].
Further place collections can be found under this path: For example, we can also list provinces registered in a country (/packages/countries/{country-name}/provinces), a districts in a province, and so on. The response would be a PlaceArray object [Listing 4.2].

GET /packages/countries/country-name/getPackageTypes List of all recognised package types in the specified country. Because legislations vary between countries, how minerals are packed, and the amount of packaging types may vary between them. This list of package types is usually read after the app gets its configuration from the getConfig controller [4.3.2].

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        country-name (String): API name of the country of interest.
```

Response 200: an array of PackageType objects [See PackageTypeResponse object on listing 4.10].
Default response: unexpected error. [See Error object on listing 4.21].

GET /packages/transports List of all packages of type 'transport' tracked by MineralTrace.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Query:
        scope (String): How deep the query should run in LDAP search scope
           format. Available values : base, one, sub
```

Response 200: an array of MineralTrace packages [See PackageResponse object on listing 4.4].
Default response: unexpected error. [See Error object on listing 4.21].

POST /packages/transports/id/doMineSale This Controller submits the order 'mine sale' to a Transport Package object. This causes its state to change from 'initialised' to 'sealed', establishing the point of origin for the minerals inside this package.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        id (String): Transport package unique ID.

    - Body: a MinesaleRequest object (application/json), OpenAPI notation on
       Listing 3.14.
```

Response 200: an updated representation of this transport package object [See PackageResponse object on listing 4.4].
Default response: unexpected error. [See Error object on listing 4.21].

POST /packages/transports/id/doSale Submits the order 'sale' to a Transport Package object. This happens when a trader resells minerals to another trader.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        id (String): Transport package unique ID.

    - Body: a SaleRequest object (application/json), OpenAPI notation on
       Listing 3.14.
```

Response 200: an updated representation of this transport package object [See PackageResponse object on listing 4.4].
Default response: unexpected error. [See Error object on listing 4.21].

GET /packages/bundles List of all packages of type 'bundle' tracked by MineralTrace.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Query:
        scope (String): How deep the query should run in LDAP search scope
           format. Available values : base, one, sub
```

Response 200: an array of MineralTrace packages [See PackageResponse object on listing 4.4].

Default response: unexpected error. [See Error object on listing 4.21].

Additionally to /packages/transports and /packages/bundles, there can be /packages/n listings of packages sorted by type, because this API supports a flexible amount of package types. On the PackageType object [4.10], the "apiURI" parameter specifies the entry point for that package type's listing.

POST /packages/bundles/id/doBundle This controller submits the order 'bundle' to a Bundle Package object. This causes its state to change from 'initialised' to 'bundled', and causes the state of all children Transport Packages put inside this bundle to also get their status set to 'bundled'.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        id (String): Transport package unique ID.

    - Body: a BundleRequest object (application/json), OpenAPI notation on
       Listing 3.14.
```

Response 200: an updated representation of this bundle package object [See PackageResponse object on listing 4.4].

Default response: unexpected error. [See Error object on listing 4.21].

POST /packages/bundles/id/doReceipt Submits the order to create a 'transport receipt' for this Bundle Package object. This happens when the bundle has arrived its intended destination as specified on the bundling process.

```
Parameters:
    - Header:
        Authorization: Bearer {JWT}

    - Path:
        id (String): Transport package unique ID.
```

```
- Body: a ReceiptRequest object (application/json), OpenAPI notation on
    Listing 3.14.
```

Response 200: an updated representation of this bundle package object [See PackageResponse object on listing 4.4].

Default response: unexpected error. [See Error object on listing 4.21].

4.3.3 Error Handling

The MineralTrace API focuses on flexiblity. Therefore, we utilise the Error object 4.21 to inform clients about any problems occurring during interaction with the server. This means, the list of error codes and their meanings can be modified at will with no repercussions to the clients. Clients should implement this abstraction and handle errors during transactions consistently; always displaying the error message to the user to ensure a good User Experience, and allowing the user to carry on their work as long as possible.

We propose an initial list of error codes that can be expanded as necessary:

Table 4.1: MineralTrace API Error Codes

Code	Description
Authentication errors	
101	General error validating a X.509 Certificate
102	Person ID does not exist
103	Person ID expired
104	Person not assigned to an Organisation
105	Organisation is invalid
106	Person / Device ID not in the same Country / Organisation
107	Person does not have enough rights to access this resource
108	Person's role not valid for this operation
Person-Related Errors	
110	Wrong role specified (Inconsistent with role on centralised database)
111	Wrong Place ID (Person not assigned to that geographical location)
112	Manual override of this person is not allowed for the current workflow
113	Not all required witnesses are present for this transaction
114	One or more participant Person IDs are invalid for this transaction
Package-Related Errors	
120	Wrong package type for this transaction
121	Package status is invalid for this transaction
122	Package is not correctly initialised
123	Package does not exist
124	Error adding child package to specified parent package
125	Weight specified is outside the tolerance limits
126	Price specified is outside the tolerance limits

Chapter 5

Conclusions

In this work, we explain why global mineral trade is vital for the high-technology industries that make our modern way or life possible, we explain what Artisanal and Small-Scale Mining (ASM) is, and discuss the grave social problems surrounding the extraction of these minerals at the source countries, including violence, human rights violations, child labour, and more. This establishes the importance of our work in the field of certification and traceability of global mineral trading chains; aiming to solve not only significant technical challenges, but also social and economic challenges affecting thousands of people around the world.

We show current international efforts to solve these social issues, such as the OECD Due Diligence Guidance for Responsible Supply Chains of Minerals from Conflict-Affected and High-Risk Areas and the Dodd-Frank legislation in the United States of America. The OECD Guidance is particularly relevant not only because it comes from a well respected international organisation, but also because it establishes a concrete ruleset and process framework for the formalisation of ASM operations. We also introduce the GOTS MineralTrace technological platform developed by German company ibes AG, and analyse how it helps governmental and private institutions to implement the OECD Guidance, its benefits and limitations. Furthermore we discuss solution alternatives to the problems identified on the current implementation of the MineralTrace platform.

We discuss possible the possible technological basis for a new version of the MineralTrace platform, including Blockchain technology, and choose the approach we consider more adequate, giving an explanation of our selection criteria.

We develop a specification for a completely new MineralTrace server-side API; which also defines a new design paradigm for MineralTrace client applications connecting to said API. The main guiding factors for the design of this new API are flexibility and law conformity. Flexibility is implemented, for example, through the concept of Workflow Definitions: The Server is now able to describe how a Trading Chain works, submit this description to a client App, and thus modify how the App behaves and captures data in order to make it compliant with the country/region where it is deployed. Furthermore, these Workflow Definitions enable the Server to also control tolerances such as the maximum allowed gold price variations, the maximum allowed sample weight variations, the personal ID types allowed for each trade step, etc. Law

conformity is implemented through the URI structure of our RESTful API. Geographical and Organisational hierarchies are respected and form the basis for the API's Resource Structure [4.1]. More importantly, in this API specification we provide solutions to all problems identified by the MineralTrace product management team.

On a practical level, our solution proposal can and should be used by the MineralTrace product management team at ibes AG to take the necessary steps to fulfil all the goals set up by the company for this platform. However, we are aware that the implementation of our recommendations represents a significant investment for the company, since many new paradigms and technologies are introduced and a significant amount of software engineering work must be invested in the practical implementation of the concepts discussed. Therefore we do not provide a practical implementation example of the redesigned MineralTrace platform in order to give ibes AG the freedom to implement it on its own terms.

On a scientific level, our technological basis choice can be considered conservative, using well-established and well-tested technologies instead of newer, more hyped technologies such as Blockchain. Our arguments against these new technologies should be viewed as basis for further scientific discussion regarding the role of Blockchain in our modern society and its feasibility as basis for solutions other than Cryptocurrencies.

Bibliography

[1] Mumtaz, A., 2015. *Natural Resource Conflicts in Afghanistan*, Thesis, Melbourne School of Land & Environment, The University of Melbourne.

[2] Spittaels, S., Hilgert, F. and International Peace Information Service (Antwerp), 2009. *Mapping Conflict Motives: Central African Republic.* IPIS.

[3] Rustad, S.A., Østby, G. and Nordås, R., 2016. *Artisanal mining, conflict, and sexual violence in Eastern DRC. The Extractive Industries and Society*, 3(2), pp.475-484.

[4] Tantalum-Niobium International Study Center, *Applications for tantalum*, URL: https://tanb.org/about-tantalum/applications-for-tantalum (accessed on May 20th, 2018), Archived at https://www.webcitation.org/5v1BooEpq on June 24th, 2013.

[5] Mineral Information Institute, *Tin*, URL: http://www.mii.org/Minerals/phototin.html (accessed on May 20th, 2018), Archived at https://www.webcitation.org/5v1C3xwLJ on December 16th, 2010.

[6] Shedd, K., 2000. *Tungsten*, United States Geological Survey.

[7] Hofmann, H., Schleper, M.C. and Blome, C., 2018. *Conflict minerals and supply chain due diligence: an exploratory study of multi-tier supply chains.* Journal of Business Ethics, 147(1), pp.115-141.

[8] OECD, 2016, *OECD Due Diligence Guidance for Responsible Supply Chains of Minerals from Conflict-Affected and High-Risk Areas: Third Edition*, URL: http://dx.doi.org/10.1787/9789264252479-en, OECD Publishing, Paris.

[9] Inc. Ibp, 2015, *Congo Democratic Republic Mineral and Mining Industry, Investment and Business Guide: Strategic Information and Basic Laws*, pp.65, Int'l Business Publications.

[10] Radley, B. and Vogel, C., 2015. *Fighting windmills in Eastern Congo? The ambiguous impact of the 'conflict minerals' movement. The Extractive industries and society*, 2(3), pp.406-410.

[11] Litvinsky, M., July 21, 2009. *DR-CONGO: Firms Fuelling 'Conflict Minerals' Violence, Report Says.*, URL: http://www.ips.org/africa/2009/07/dr-congo-firms-fuelling-conflict-minerals-violence-report-says, Inter Press Service, Archived at https://www.webcitation.org/5v1CoG37B on December 16th, 2010.

[12] Hentschel, T., 2003. *Artisanal and small-scale mining: challenges and opportunities.* Iied.

[13] Labonne, B. and Gilman, J., 1999. *Towards Building Sustainable Livelihoods in the Artisanal Mining Communities.*, Paper presented at the Tripartite Meeting on Social and Labour Issues in Small- Scale Mines, 17–21 May, Geneva

[14] Geneva: International Electrotechnical Commission., 2001, *IEC 60529, "Degrees of Protection Provided by Enclosures (IP Codes)", ed. 2.1*

[15] Shedd, Kim B., 2006. *2006 Minerals Yearbook: Cobalt.*, URL: https://minerals.usgs.gov/minerals/pubs/commodity/cobalt/myb1-2006-cobal.pdf (accessed on May 30th, 2018), United States Geological Survey.

[16] Nakamoto, S., *Bitcoin: A peer-to-peer electronic cash system.* Whitepaper, 2008.

[17] Tapscott, D. and Tapscott, A., 2016. *Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world.* Penguin.

[18] Chaum, D., 1983. *Blind signatures for untraceable payments.* Advances in Cryptology Proceedings, 82 (3): 199–203.

[19] Chaum, D. et al., 1992. *Achieving Electronic Privacy.* Scientific American, 267, 96-101.

[20] Szabo, N., December 27, 2008. *Bit gold.*, URL: http://unenumerated.blogspot.com/2005/12/bit-gold.html (accessed on June 3rd, 2018), Unenumerated.

[21] Biggs, J., Hinish, S.R., Natale, M.A. and Patronick, M., *Blockchain: Revolutionizing the Global Supply Chain by Building Trust and Transparency.*

[22] Weber, M., 1965. *Politics as a Vocation.*

[23] Weber, M., 1918. *Weber's Rationalism and Modern Society, translated and edited by Tony Waters and Dagmar Waters.* Palgrave Books.

[24] Olson, M., 1993. *Dictatorship, democracy, and development.* American political science review, 87(3), pp.567-576.

[25] Dölle, M., 2018. *Wie abgeschlossene Transaktionen aus der Blockchain verschwinden.* c't Magazin, URL: https://www.heise.de/ct/ausgabe/2018-8-Wie-abgeschlossene-Transaktionen-aus-der-Blockchain-verschwinden-4004027.html (accessed on June 5th, 2018).

[26] Apodaca, R. 2017. *How to Clear a Stuck Bitcoin Transaction.* Bitzuma.com. URL: https://bitzuma.com/posts/how-to-clear-a-stuck-bitcoin-transaction (accessed on June 5th, 2018).

[27] Deetman, S. 2016. *Bitcoin Could Consume as Much Electricity as Denmark by 2020.* VICE. URL: https://motherboard.vice.com/en_us/article/aek3za/bitcoin-could-consume-as-much-electricity-as-denmark-by-2020 (accessed on June 5th, 2018).

[28] Greenfield, R. 2017. *Vulnerability: Proof of Work vs. Proof of Stake.* MEDIUM. URL: https://medium.com/@robertgreenfieldiv/vulnerability-proof-of-work-vs-proof-of-stake-f0c44807d18c (accessed on June 4th, 2018).

[29] Chuen, D. L. K. ed., 2015. *Handbook of digital currency: Bitcoin, innovation, financial instruments, and big data.* Academic Press; May 5 2015.

[30] Loukides, M., 2018. *Steering around blockchain hype.* O'Reilly Media. URL: https://www.oreilly.com/ideas/steering-around-blockchain-hype (accessed on June 6th, 2018)

[31] Regulation, G.D.P., 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46. Official Journal of the European Union (OJ), 59, pp.1-88.

[32] Wagh, K. and Thool, R., 2012. *A comparative study of soap vs rest web services provisioning techniques for mobile host.* Journal of Information Engineering and Applications, 2(5), pp.12-16.

[33] Fielding, R. T., 2000. *Architectural Styles and the Design of Network-Based Software Architectures.* Doctoral Dissertation. Dept. of Computer Science, Univ. of California.

[34] Feng, X., Shen, J. and Fan, Y., 2009, October. *REST: An alternative to RPC for Web services architecture.* In Future Information Networks, 2009. ICFIN 2009. First International Conference on (pp. 7-10). IEEE.

[35] Maurer, U., 1996, September. *Modelling a public-key infrastructure.* In European Symposium on Research in Computer Security (pp. 325-350). Springer, Berlin, Heidelberg.

[36] Cooper, D., 2008. *Internet X. 509 public key infrastructure certificate and certificate revocation list (CRL) profile.* RFC 5280.

[37] Masse, M., 2011. *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces.* O'Reilly Media, Inc.

[38] Jones, M., Bradley, J. and Sakimura, N., 2015. Json web token (jwt) (No. RFC 7519).

[39] Jones, M. and Hildebrand, J., 2015. Json web encryption (jwe) (No. RFC 7516).

[40] *Manuel des procedures de traçabilité des produits miniers: de l'extraction a l'exportation.* 2ème Edition. Democratic Republic of the Congo. URL: http://www.leganet.cd/Legislation/Droit%20economique/Code%20Minier/Manueldesprocedures2014.pdf (accessed on July 2nd, 2018).

Declaration of Authorship

I hereby declare that I am the sole author of this bachelor thesis and that I have not used any sources other than those listed in the bibliography and identified as references. I further declare that I have not submitted this thesis at any other institution in order to obtain a degree.

Chemnitz, August 21, 2018.

José Ernesto Olmedo Pereira.